Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Electrónica

Licenciatura en Electrónica

DESARROLLO DE UN SISTEMA DE RECONOCIMIENTO DE UN MOVIMIENTO PARA LA EMISIÓN DE UNA ALERTA.

TESIS
Para obtener el título de:
Licenciado en Electrónica
PRESENTA:

C. Jaime Yovani Pérez Romero

Directores de Tesis: **Dra. María Monserrat Morín Castillo (FCE - BUAP)**
Dr. José Rubén Conde Sánchez (FCFM - BUAP)

Agosto 2024. Puebla, Pue. Méx.

Resumen

Las tendencias tecnológicas han avanzado a la par con las cambiantes necesidades de sus usuarios, y en el ámbito del hogar, es de manera similar. Esta tendencia ha impulsado estudios, aplicaciones y desarrollos en la domótica enfocada en la automatización inteligente en viviendas y edificios, ofreciendo servicios que cubren desde emisión de alertas, la seguridad, la gestión energética, el confort hasta la comunicación, entre otros. En el ámbito de la seguridad, se han adoptado estrategias de automatización que emplean interfaces de voz o pantallas táctiles. La presente tesis propone el estudio y desglose de un sistema de reconocimiento de movimiento de una persona através de sus extremidades para la emisión de una alerta, el uso de la plataforma *Neuton ai* para el entrenamiento de un algoritmo clasficador binario y la implementación del algoritmo en dispositivos tipo tinyML. La aplicación del sistema planteado puede asociarse a diversas posibilidades para el control eficiente de dispositivos domésticos, por ejemplo: emisión de alertas, gestión de la iluminación, control persianas o cerraduras utilizando movimientos específicos. La orientación de la tesis es hacia personas que lo requieran para brindar un recurso tecnológico en su vida diaria.

Palabras Clave: Domótica, seguridad, aprendizaje automático.

Índice general

Resumen I

Agradecimientos I

1. Introducción **1**
1.1. Planteamiento del problema 1
1.2. Motivación 2
1.3. Justificación 2
1.4. Objetivos 4
1.4.1. Objetivo general 4
1.4.2. Objetivos particulares 4
1.5. Diagrama metodológico 4
1.6. Organización de la tesis 6

2. Marco teórico **7**
2.1. Antecedentes de la domótica 7
2.2. Dispositivos utilizados en la domótica 8
2.2.1. Sensores 8
2.2.2. Motores 11
2.2.3. Actuadores 12
2.2.4. Dimmers y relés 12
2.2.5. Acondicionador de señal 13
2.3. Inteligencia artificial 14
2.3.1. Aprendizaje automático 15
2.3.2. Aprendizaje profundo 15
2.3.3. Aprendizaje automático TinyML 16
2.3.4. Redes neuronales artificiales 16
2.3.5. Modelo simplificado de una red neuronal artificial 17
2.4. Componentes de una red neuronal artificial 19
2.5. Estado del arte 21

3. Componentes básicos para el sistema **25**
3.1. Gestos y movimientos corporales 25
3.1.1. Movimientos 25
3.1.2. Tipos de movimientos 26
3.1.3. Gestos 26
3.1.4. Tipos de gestos 26
3.2. Elección de movimientos 27
3.3. Reconocimiento de movimientos 28
3.3.1. Gestiones de la domótica que pueden ser controlados mediante movimientos 28
3.3.2. La captación de movimientos 30
3.3.3. Sensores (giroscopios y acelerómetros) 30
3.3.4. Captación de movimientos con la Nicla Sense ME 31
3.4. Apredizaje automático TinyML 32
3.4.1. TinyML y como se diferencia del aprendizaje automático tradicional 33
3.4.2. Aplicaciones prácticas de TinyML en la vida cotidiana 34
3.4.3. Problemas más comunes al desarrollar modelos de TinyML 34
3.5. Microcontroladores y procesadores 35
3.5.1. Microcontroladores para ML 36
3.5.2. Microcontroladores para TinyML 37
3.6. Plataformas para programar microcontroladores 37
3.7. Plataformas para desarrollar modelos ML 38
3.7.1. Algoritmos de clasificación 41
3.7.2. Algoritmos de regresión 41
3.7.3. Aplicación en el sistema de reconocimiento de movimientos 41
3.7.4. Tabla comparativa de algoritmos 42
3.8. Teoría matemática sobre la regresión lineal 42
3.8.1. Regresión lineal simple 42
3.8.2. Cálculo de los coeficientes 43
3.8.3. Evaluación del modelo 43
3.8.4. Regresión Lineal en Neuton TinyML 44

4. Reconocimiento de movimiento **45**
4.1. Fase 1: Adquisición de los datos obtenidos por acelerómetro y giroscopio 45
4.2. Fase 2: Capturar datos de entrenamiento y almacenamiento en formato CSV con codificación UTF-8 o ISO-8859-1 46
4.3. Fase 3: Entrenamiento del modelo 47
4.3.1. Fase 4: Entrenar algoritmo de clasificación binaria para la detección del movimiento propuesto 49
4.3.2. Fase 5: Construir el enlace entre el reconocimiento del movimiento y la emisión de la alerta de ayuda 52
4.4. Razones para la elección de alertas por correo electrónico 55

4.5. Alertas alternativas al correo electrónico 55
4.5.1. Notificaciones por WhatsApp 55
4.5.2. Mensajes de texto (SMS) 56
4.5.3. Activación de una alarma sonora 56

5. Resultados **57**
5.0.1. Resultados de pruebas de detección 58
5.0.2. Frecuencia de muestreo 59
5.0.3. Comparación con otras frecuencias de muestreo 59
5.0.4. Análisis Grafico 59
5.0.5. Implementación y validación 61

Agradecimientos

Quiero expresar mi más profundo agradecimiento a mis padres por su esfuerzo y paciencia, cuyo apoyo incondicional y sacrificio han sido fundamentales para la realización de esta tesis. Sin su constante aliento y amor, este logro no hubiera sido posible.

Asimismo, deseo agradecer a la Dra. María Monserrat Morín Castillo y al Dr. José Rubén Conde Sánchez por sus valiosos consejos y su constante apoyo durante el desarrollo de esta tesis. Su guía académica y profesional ha sido esencial para la culminación exitosa de este proyecto.

A todos, muchas gracias.

Índice de figuras

1.1. Herramientas propuestas de accesibilidad en el aprendizaje automático [5] 3
1.2. Diagrama metodológico propuesto. 5

2.1. Sensores de temperatura [12]. 9
2.2. Sensores de iluminación [13]. 10
2.3. Sensores de seguridad. 10
2.4. Diferentes tipos de motores [14]. 11
2.5. Actuadores. 12
2.6. Acondicionadores de señal. 13
2.7. Red neuronal natural (izquierda), artificial (derecha)[27]. 17
2.8. Partes fundamentales de una ANN [31] 18
2.9. Entradas a la red neuronal. 19
2.10. Pesos W (coeficientes). 19
2.11. Bias B. 20
2.12. Transferencia F. 20
2.13. Relojes inteligentes. 21
2.14. Sensores de Tejido de Punto [37]. 22
2.15. Camaras de reconocimiento de acciones humanas [39]. 22
2.16. Camaras de reconocimiento de acciones humanas [41]. 23
2.17. Sensores integrados en equipos de protección personal utilizados en fábricas [43]. 23

3.1. Microprocesador. 36
3.2. Microcontrolador. 36
3.3. Plataformas para programar microcontroladores. 38
3.4. Edge impulse [50]. 39
3.5. Espressif [47]. 39
3.6. TensorFlow [52]. 39
3.7. Neuton tiny ML [53]. 39
3.8. Regresión lineal [56]. 44

4.1. Conexión Nicla Sense ME. 46
4.2. Seleccionar archivo de datos en *Neuton*. 48

4.3. Especificación del conjunto de datos. 49

5.1. Movimiento de brazo Flexión-Extensión 57
5.2. Movimiento de mano Flexión-Extensión. 57
5.3. Entrenamiento del movimiento Caso I: BRAZO 60
5.4. Entrenamiento del movimiento Caso II: MANO 60
5.5. Serial monitor y Serial Plotter. 61
5.6. Análisis de movimiento Caso I: BRAZO 62
5.7. Validación y reconocimiento del movimiento Caso I. 62
5.8. Análisis de movimiento Caso II: MANO 63
5.9. Validación y reconocimiento del movimiento Caso 2. 63

Índice de tablas

3.1. Gestión de Energía. 28
3.2. Gestión de Confort. 29
3.3. Gestión de Seguridad. 29
3.4. Descripción de los sensores LSM6DS33TR, MPU6050 y Nicla Sense ME. 31
3.5. Microcontroladores para aprendizaje automático TinyML. 37
3.6. Comparación de plataformas para entrenar modelos de aprendizaje automático. 40
3.7. Comparación de algoritmos de clasificación y regresión. 42

5.1. Resultados de evaluación del modelo para movimientos de brazo y mano con diferentes sensores. 58
5.2. Comparación de frecuencias de muestreo 59

Capítulo 1

Introducción

última década, la domótica ha emergido como un campo tecnológico que busca mejorar la calidad de vida de las personas a través de la automatización y el control inteligente de sus hogares [1]. Uno de los aspectos más destacados de la domótica es su capacidad para mejorar la seguridad del hogar mediante la implementación de sistemas de alerta y respuesta rápida ante situaciones de emergencia [2].

Algunos desarrollos están centralizados en sistemas que identifican movimientos de las extremidades superiores o inferiores en personas que presentan alguna discapacidad que limita su movimiento o su habla. Además, se han implementado tecnologías avanzadas que integran sensores y algoritmos de inteligencia artificial para mejorar la precisión en la detección de estos movimientos. Estos sistemas no solo facilitan la comunicación y la interacción de las personas con discapacidades, sino que también proporcionan una mayor independencia y calidad de vida al permitir un control más efectivo sobre dispositivos y entornos digitales [3].

Por otro lado, el avance de la inteligencia artificial en los sistemas embebidos ha generado un consenso favorable en su uso dentro de la comunidad de sistemas integrados sobre la necesidad de soportar el aprendizaje automático. Con la creciente demanda de sistemas domóticos y dispositivos IoT, se ha vuelto importante proporcionar un valor adicional al usuario a través de soluciones basadas en plataformas de diseño online [4]. Así, surge la necesidad de integrar elementos sensoriales en paquetes ultra pequeños, llevando al extremo la capacidad de los dispositivos como elementos de detección [4, 1, 2, 3].

1.1. Planteamiento del problema

Aunque se han logrado avances en la domótica, la mayoría de los sistemas de seguridad disponibles están fuera del alcance de satisfacer necesidades específicas de personas con discapacidades motoras, del habla o parálisis de algún tipo. Estas personas enfrentan desafíos adicionales para acceder a medidas de seguridad convencionales, lo que las coloca en situaciones vulnerables.

La limitación para activar señalizaciones de atención de manera rápida y efectiva puede resultar en retrasos en la atención, aunque no necesariamente médica, que en caso de serlo puede aumentar el riesgo de lesiones o complicaciones para estas personas. La mayoría de los sistemas de seguridad existentes requieren algún tipo de interacción, ya sea manual o verbal, para activarse, lo cual dificulta o imposibilita a las personas con discapacidades motoras y del habla. Esto crea un espacio de oportunidad para crear accesos a la seguridad para este grupo de personas, que requiere soluciones tecnológicas para proteger su bienestar en el hogar.

Desde el punto de vista del desarrollo tecnológico, el reto es crear sistemas de seguridad en el hogar que sean verdaderamente inclusivos y accesibles para personas con discapacidades motoras y del habla. Aunque se ha avanzado en tecnologías de automatización y reconocimiento de movimiento, faltan soluciones específicas que realmente cubran las necesidades de estas personas. Muchos de los dispositivos actuales no consideran la variedad de limitaciones físicas y de comunicación, por lo que requieren interacciones manuales o verbales que no todos pueden realizar. Esto destaca la necesidad urgente de desarrollar tecnologías adaptativas que utilicen interfaces no tradicionales, como el reconocimiento de gestos y movimientos, para activar sistemas de seguridad de manera rápida y eficiente.

Para el desarrollo de la tesis se propone utilizar Neuton.ai [4], una herramienta destacada por su capacidad para automatizar por completo el proceso de diseño, optimización y despliegue de modelos de aprendizaje automático en dispositivos de tamaño reducido. La principal razón para elegir Neuton.ai es su habilidad para permitir desarrollar aprendizaje automático a través de la creación de arquitecturas de redes neuronales adaptadas específicamente para aplicaciones de TinyML de manera automatizada, simplificando y acelerando significativamente el desarrollo de soluciones de inteligencia artificial en el borde de la red.

1.2. Motivación

Una de las principales motivaciones para realizar la tesis sobre esta temática es el interés de contribuir y desarrollar tecnologías de seguridad en el hogar que sean inclusivas y accesibles para personas con discapacidades motoras y del habla. Aunque ya existen avances tecnológicos en este sentido, existen oportunidades importantes para contribuir con soluciones adaptadas a las necesidades de estas personas.

1.3. Justificación

La justificación para la presente tesis radica en realizar la implementación de un sistema de alerta para personas con cierto tipo de discapacidad, como aquellas que se encuentran postradas en cama y presentan una mínima movilidad, como las descritas en la introducción 1. Según un informe de la Organización Mundial de la Salud de 2011, se

estimaba que más de mil millones de personas, incluidos los niños (o alrededor del 15 % de la población mundial), vivían con algún tipo de discapacidad. Se proyecta que esta cifra aumentará a más de dos mil millones para 2050 a medida que la población mundial envejezca y aumente la prevalencia de enfermedades no transmisibles [5] .

Por lo general, las personas con discapacidad experimentan peores niveles de salud, menos logros educativos, menos oportunidades económicas y tasas de pobreza más altas que las personas sin discapacidad [5]. Es por eso que el desarrollo de un sistema como el propuesto impactaría de forma positiva a las personas con dificultades motoras o problemas del habla y permitiría obtener ventajas como:

- **Accesibilidad mejorada:** Un sistema de reconocimiento de movimientos elimina la necesidad de manipular interruptores físicos o dispositivos de control, facilitando la gestión domótica para las personas con alguna discapacidad.
- **Independencia reforzada:** Permite a las personas controlar las funciones del hogar de manera autónoma y sin necesidad de ayuda adicional, fomentando la autonomía y la autoconfianza.

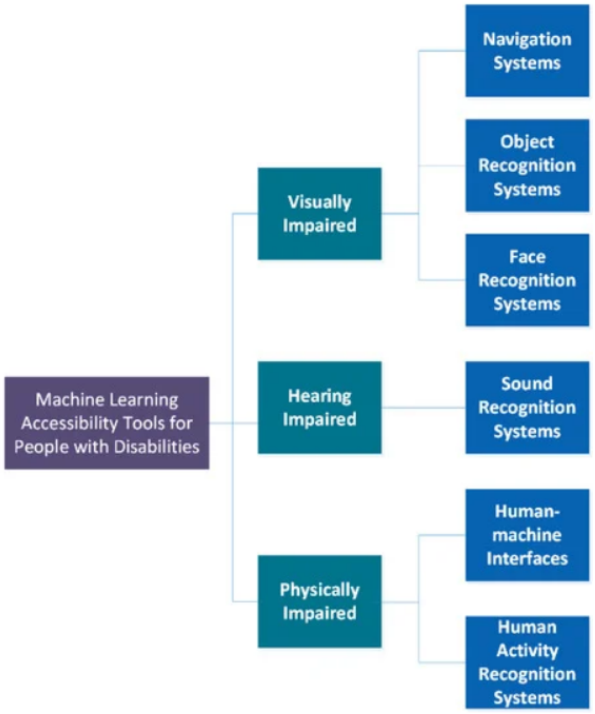


Figura 1.1: Herramientas propuestas de accesibilidad en el aprendizaje automático [5]

Se plantea un desarrollo atractivo, con diseños que requieran poco tiempo para planearse, desarrollarse, implementarse y ponerse a prueba. Por esta razón, se hará uso de plataformas en línea que permitan involucrar diferentes fases de diseño tradicionales en

formas ágiles de entrenar algoritmos de inteligencia artificial y que permitan la implementación de los algoritmos en dispositivos programables de bajo consumo de energía, aprovechando esta valiosa ventaja.

La automatización que ofrece Neuton.ai elimina la necesidad de realizar búsquedas manuales de arquitecturas de redes neuronales y optimización de modelos, lo que no solo ahorra tiempo y reduce la complejidad, sino que también permite a los desarrolladores centrarse en la creación de aplicaciones en lugar de tareas tardadas de diseño y ajuste de modelos. Además, la capacidad de Neuton.ai para generar modelos de redes neuronales eficientes en términos de consumo de energía. También se plantea la tecnología TinyML [6], que permite implementar modelos de aprendizaje profundo en dispositivos IoT de baja potencia. Al procesar datos localmente en el dispositivo, se reduce la latencia, se optimizan los recursos disponibles y disminuyen los costos asociados con el procesamiento en la nube. Además, su eficiencia energética lo hace adecuado para dispositivos con limitaciones de energía. Con aplicaciones diversas en campos como el reconocimiento de gestos, voz y vehículos autónomos, TinyML ofrece una solución versátil y eficaz para llevar la inteligencia artificial a entornos con recursos limitados.

1.4. Objetivos

1.4.1. Objetivo general

Desarrollar un sistema que identifique un movimiento binario en una persona para posteriormente emitir una alerta y que alguien acuda a proporcionarle atención.

1.4.2. Objetivos particulares

- Utilizar la plataforma Neuton AI para entrenar el modelo de aprendizaje automático.
- Seleccionar y entrenar el algoritmo de clasificación binaria para que pueda reconocer el movimiento establecido.
- Seleccionar el sensor de movimiento y el microcontrolador adecuado para el sistema.
- Implementar el algoritmo de clasificación binaria para su validación y pruebas.

1.5. Diagrama metodológico

Esta tesis, que tiene como objetivo principal desarrollar un sistema que identifique un movimiento binario de una persona para emitir una alerta y así brindar la atención necesaria, se aborda con la siguiente metodología propuesta, basada en 5 fases principales, para asegurar el éxito del objetivo general:

- **Inicio:** Se realiza un movimiento binario ya establecido.

- **Adquisición y procesamiento de datos:** Este proceso utiliza un sensor para capturar datos sobre el movimiento realizado por el usuario. Este sensor puede ser un acelerómetro, un giroscopio o un sensor de movimiento. Al capturar los datos, el sensor los envía al microcontrolador para su ordenamiento y procesamiento. El microcontrolador debe ser un dispositivo programable compatible con la plataforma Neuton ML.

- **Entrenamiento del modelo de reconocimiento de un movimiento:** Utilizando la plataforma Neuton ML, se entrena un modelo de aprendizaje automático para reconocer el gesto binario corporal específico que se desea controlar. Para ello, se utilizan conjuntos de datos etiquetados que contienen ejemplos de movimientos corporales y las acciones asociadas al movimiento.

- **Implementación del modelo en un dispositivo programable:** Una vez que el modelo de reconocimiento de movimiento ha sido entrenado, se implementa en un dispositivo programable.

- **Final:** El sistema debe estar conectado a la alerta que se desea activar por un movimiento. Esto puede implicar la integración con sistemas de automatización del hogar compatibles con Neuton ML como Wi-Fi o Bluetooth.

En la Figura 1.2, se muestran las fases de forma esquemática.

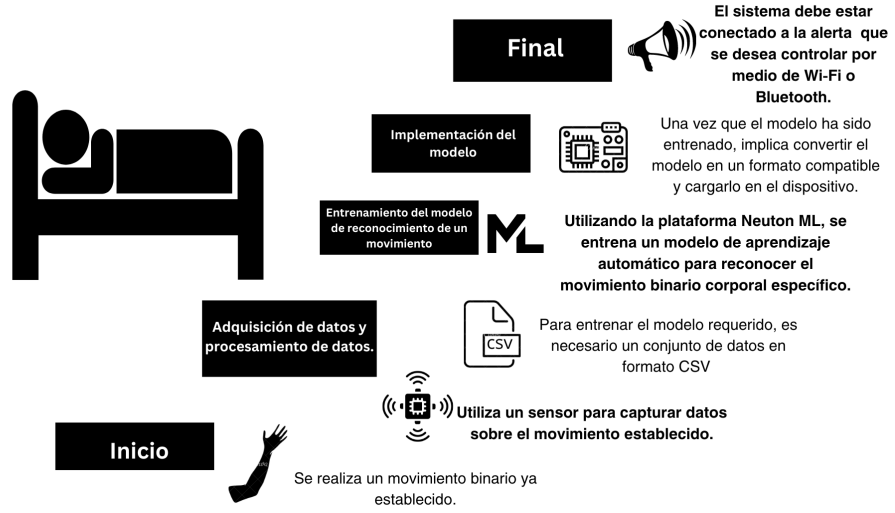


Figura 1.2: Diagrama metodológico propuesto.

1.6. Organización de la tesis

La presente tesis se encuentra organizada en cinco capítulos, cuyo contenido se describe a continuación:

- Capítulo 1: Se presenta la introducción, justificación y objetivos sobre el sistema de reconocimiento de un movimiento para la emisión de una alerta.
- Capítulo 2: Se presenta el marco teórico y estado del arte de algunos sistemas similares.
- Capítulo 3: Se presentan algunas definiciones y elementos básicos que permiten seguir el trabajo desarrollado.
- Capítulo 4: Se enfoca en presentar el desarrollo de un sistema de reconocimiento de un movimiento.
- Capítulo 5: Se presentan los principales resultados del desarrollo del sistema y conclusiones obtenidas.

Capítulo 2

Marco teórico

2.1. Antecedentes de la domótica

En la década de los 60 y 70, al igual que con el desarrollo de la sociedad, la ciencia, el arte, las humanidades y en particular con el avance de la tecnología y dado auge de los electrodomésticos, surgen las primeras muestras de hogares inteligentes, muestra de esto es el protocolo de comunicación X10, el cual era un protocolo de comunicación para el control doméstico remoto; y, si bien fue propuesto e inicialmente desarrollado por Pico Electronics of Glenrothes, Escocia [7], este fue introducido en los hogares en la década de los 70's por los Estados Unidos. Este primer sistema dómotico ya permitía la comunicación entre electrodomésticos y sistemas eléctricos en el hogar a través del cableado eléctrico existente en la vivienda [8]. El protocolo de comunicación X10 operaba mediante señales codificadas superpuestas a la corriente alterna doméstica, enviadas en forma de pulsos sincronizados con el cruce por cero de la corriente. Los comandos eran encendido o apagado, ajuste de brillo y consultas de estado de algún dispositivo. Los transmisores X10 generaban las señales de control, mientras que los receptores las interpretan y ejecutan las acciones correspondientes.

La idea de un hogar automático ha evolucionado significativamente a lo largo de los años. Aunque el concepto de domótica comenzó a tomar forma en los años 90 y principios de los 2000, fue el avance de la tecnología de la información, la conectividad inalámbrica y móvil lo que llevó a la domótica a un nuevo nivel. Esto permitió la creación de redes domésticas que podían ser controladas remotamente, revolucionando la forma en que las personas interactúan con sus hogares. Donta Praveen en su artículo en 2022 [9] destaca el crecimiento del Internet de las Cosas (IoT) en los últimos años, mostrando su flexibilidad y facilidad de uso en aplicaciones en tiempo real. Donta menciona la importancia de los protocolos de la capa de aplicación en el IoT y se plantea la posibilidad de integrar el *aprendizaje automático* en este campo para impulsar nuevas direcciones de investigación.

El desarrollo y difusión de IoT va llevando a la domótica a nuevas oportunidades, permitiendo que los dispositivos y sistemas del hogar no sólo se controlen de forma remota, sino que también interactuen entre sí, aprendan de los patrones de comportamiento

del usuario y se adapten automáticamente a sus preferencias.

Hoy en día, los sistemas domóticos pueden controlar casi todos los aspectos del entorno de una casa, desde la iluminación y la temperatura, hasta la seguridad y el entretenimiento, proporcionando mayor comodidad, eficiencia energética y seguridad. Dada su evolución, de estos sistemas, actualmente se integran el uso de la inteligencia artificial y el aprendizaje automático. Wenda Li en un estudio realizado para la University Reading en 2022 [10] detalla que el concepto de hogar inteligente se ha expandido a lo largo de los años, abarcando diversas áreas de aplicación. Por ejemplo, en el ámbito de la energía doméstica, los hogares inteligentes permiten el monitoreo y la gestión remota para optimizar el consumo energético, mejorando así la eficiencia y reduciendo el gasto. Wenda menciona que en el cuidado de la salud se ofrecen asistencias en casa a través de dispositivos inteligentes, facilitando la toma de decisiones y brindando apoyo diario a los usuarios. Además, menciona que el concepto de hogar inteligente se aplica para crear entornos de vida agradables, seguros y estéticamente elegantes, mejorando la calidad de vida de los residentes. A pesar de su amplia adopción en diversas áreas, aún no existe un consenso claro sobre lo que realmente representa el hogar inteligente y cuáles serán sus tendencias futuras. La investigación destaca la importancia de proporcionar un entorno seguro y saludable, mejorar el bienestar de los ocupantes y contribuir a la prestación de servicios de atención médica.

Asi, es relevante considerar la importancia de los sensores y dispositivos utilizados para las diversas aplicaciones que pudieran operar o no en tiempo real, esto requiere flexibilidad y facilidad de uso de estos dispositivos en el contexto de la creciente adopción de la domótica apoyada en la IoT.

2.2. Dispositivos utilizados en la domótica

2.2.1. Sensores

Los sensores son dispositivos electrónicos diseñados para detectar cambios en el entorno y convertirlos en señales eléctricas o digitales que pueden ser interpretadas por otros dispositivos o sistemas. En el campo de la domótica, los sensores juegan un papel fundamental al detectar condiciones específicas del entorno del hogar y transmitir esa información a los sistemas de control para realizar acciones automatizadas.

Existen una variedad de sensores utilizados los cuales se pueden clasificar en cuatro categorías principales. Una de estas categorías son los sensores de climatización o temperatura, que miden la temperatura y la humedad del ambiente y permiten regular la calefacción, el aire acondicionado, las persianas o el riego automático según las condiciones climáticas externas o las preferencias de los usuarios.

Algunos de estos sensores son: *Sensor de temperatura ambiente* que mide la temperatura del aire en un espacio determinado, son utilizados comúnmente en termostatos, sistemas de calefacción, ventilación y aire acondicionado y otros sistemas de control de

temperatura. *Sensor infrarrojo* que utiliza la tecnología infrarroja para medir la temperatura de un objeto o superficie sin contacto directo. Estos sensores se utilizan en aplicaciones como termómetros sin contacto, sistemas de detección de temperatura corporal o control de temperatura en procesos industriales. *Sensor de humedad* mide el nivel de humedad relativa en el ambiente. Se utiliza junto con los sensores de temperatura para controlar y regular la climatización de un espacio. Y, *Sensor de presión barométrica* mide la presión atmosférica y se utiliza para calcular la altitud y predecir cambios climáticos. Estos sensores son comunes en estaciones meteorológicas y sistemas de control de clima [11].

En la Figura 2.1 se muestran algunos tipos de sensores de temperatura.

Figura 2.1: Sensores de temperatura [12].

En la domótica, además de los sensores de climatización, otro grupo esencial lo constituyen los sensores de iluminación. Estos dispositivos desempeñan un papel crucial en la gestión eficiente y automatizada del sistema de iluminación del hogar, mejorando tanto la comodidad como el ahorro energético [13]. Algunos de estos sensores son: *Sensores de iluminaciòn* que miden la intensidad y el color de la luz y permiten encender o apagar las luces, también cambiar sus tonalidades o brillo según la hora del día. *Sensor de luz ambiental* mide la cantidad de luz existente en un área y ajusta la iluminación artificial en función de esta información. *Sensor de movimiento* que detecta el movimiento en un área determinada y enciende o apaga la iluminación en respuesta a la presencia de personas. *Sensor de ocupación* que es similar al sensor de movimiento, pero utilizado específicamente para detectar si una habitación o área está ocupada, comúnmente en entornos comerciales y públicos para ahorrar energía apagando automáticamente las luces cuando no hay nadie presente. *Sensor de crepúsculo* que controla la iluminación exterior en función de la cantidad de luz natural disponible, encendiendo las luces al anochecer y apagándolas al amanecer [13].

En la Figura 2.2 se muestran algunos tipos de sensores de iluminación.

Figura 2.2: Sensores de iluminación [13].

En el ámbito de la domótica, la seguridad del hogar es una prioridad fundamental que se aborda mediante la integración de diversos tipos de sensores diseñados para detectar y responder a situaciones de riesgo [13]. Estos sensores no solo proporcionan alertas tempranas para proteger a los ocupantes y la propiedad, sino que también pueden tomar medidas preventivas automáticas para mitigar posibles daños. A continuación, se presentan algunos de los sensores de seguridad más comunes y sus funciones específicas. Los sensores de seguridad son sensores que detectan situaciones de riesgo como incendios, fugas de agua o gas, roturas de cristales o intrusiones y permiten alertar al usuario o a los servicios de emergencia, así como tomar medidas preventivas como cortar el suministro de agua o gas, activar sistemas antiincendios o cerrar puertas y ventanas. Algunos ejemplos de estos sensores son: *Sensor de movimiento*, que detecta cualquier movimiento dentro del área de cobertura y activa una alarma o enciende las luces de seguridad; *Sensor de puerta o ventana*, que detecta si una puerta o ventana se abre o se cierra de manera no autorizada, siendo ampliamente utilizados en sistemas de seguridad residenciales y comerciales; *Sensor de rotura de vidrios*, que detecta el sonido característico de un vidrio rompiéndose y dispara una alarma, protegiendo ventanas y puertas de vidrio; *Sensor de humo*, que detecta el humo en caso de incendio y activa las alarmas de seguridad correspondientes, esenciales en los sistemas de seguridad contra incendios; y *Sensor de monóxido de carbono*, que mide los niveles de monóxido de carbono, un gas peligroso, y activa las alarmas cuando se alcanzan niveles peligrosos, vitales para proteger la salud y la seguridad en ambientes interiores.

En la Figura 2.3 se muestran algunos tipos de sensores de seguridad.

Figura 2.3: Sensores de seguridad.

2.2.2. Motores

Los motores se consideran parte del grupo de actuadores, generan fuerza mecánica capaz de desplazar o rotar objetos, como: persianas, toldos, puertas, ventanas entre otros. Existen diferentes tipos de motores y estan clasificados según el tipo de corriente que utilizan, el tipo de movimiento que generan o el tipo de control que tienen. Algunos de estos tipos motores son:

Los *motores de corriente continua* que operan con corriente eléctrica constante y unidireccional, estos se pueden usar para aplicaciones que requieren precisión y velocidad variable, como los robots, los drones o los coches eléctricos.

Los *motores de corriente alterna* que operan con una corriente elétrica cambiante y periorida, estos se pueden usar para aplicaciones que requieren potencia y velocidad constante, como los ventiladores, las bombas o los electrodomésticos.

Los *motores paso a paso* son motores compuestos de varios imánes o bobinas que se activan secuencialmente para generar un movimiento angular discreto y preciso, son usados para aplicaciones que requieren precisión y control, como las impresoras 3D, las cámaras o los relojes.

Los *motores servo* son motores que se componen de un motor de corriente continua, un reductor de velocidad y un sensor de posición. Se usan para aplicaciones que requieren precisión y fuerza, como los brazos robóticos, las antenas o los drones [14].

Figura 2.4: Diferentes tipos de motores [14].

2.2.3. Actuadores

Los actuadores son responsables de ejecutar acciones en respuesta a los datos recopilados por los sensores, ver Figura 2.5. Algunos tipos de actuadores y aplicaciones incluyen: reguladores de intensidad para ajustar la iluminación según las preferencias del usuario; relés para encender o apagar luces de forma remota; termostatos inteligentes que regulan la temperatura del hogar de manera eficiente; válvulas motorizadas que controlan el flujo de agua en sistemas de calefacción radiante; cerraduras electrónicas que permiten el acceso mediante códigos, tarjetas o reconocimiento biométrico; motores para persianas o cortinas que se cierran o abren automáticamente; sistemas de audio y video que se controlan de forma remota; enchufes inteligentes que permiten encender o apagar dispositivos conectados a la corriente eléctrica; válvulas solenoides que controlan el flujo de agua en sistemas de riego automatizados; y motores para ventanas automáticas que regulan la ventilación en función de la calidad del aire o la temperatura interior. Estos actuadores ejecutan acciones como encender las luces, abrir cortinas o activar sistemas de climatización, mejorando la comodidad, la seguridad y la eficiencia energética en el hogar [13].

Figura 2.5: Actuadores.

El control de la iluminación, actuadores, motores y de otros dispositivos eléctricos es esencial para mejorar la eficiencia energética. Los dimmers y los relés son otros componentes fundamentales en estos sistemas, ya que permiten una gestión precisa de la luz y otros aparatos eléctricos, los cuales serán descritos a continuación.

2.2.4. Dimmers y relés

Los dimmers son dispositivos que permiten regular la intensidad de la luz artificial en un ambiente, creando diferentes escenas y ahorrando energía. Los dimmers se clasifican según el tipo de tecnología que utilizan para regular la luz, como el corte de fase, el PWM, el 1-10V o el DALI. Los dimmers se pueden controlar de forma manual o automática, mediante interruptores, mandos a distancia, sensores o aplicaciones móviles [15].

Los relés son actuadores que permiten abrir o cerrar un circuito eléctrico mediante un electroimán, lo que permite encender o apagar dispositivos como luces, enchufes, electrodomésticos, etc. Existen diferentes tipos de relés usados como actuadores en la domótica, según el tipo de corriente que utilizan [16].

La precisión y fiabilidad de la información que manejan los sistemas inteligentes son cruciales para un adecuado funcionamiento. Para asegurarse que los datos obtenidos de

los diversos sensores y actuadores sean adecuados para su procesamiento y digitalización, se emplean acondicionadores de señal.

2.2.5. Acondicionador de señal

El acondicionador de señal es un sistema electrónico contenido en dispositivo que convierte un tipo de señal analógica o digital en otro tipo de señal adecuada para su procesamiento o digitalización. La Figura 2.6 muestra una representación de su funcionamiento. Los acondicionadores de señal se utilizan para adaptar las señales que proporcionan los sensores y los actuadores al sistema inteligente que controla una vivienda [13].

Funciones de los acondicionadores de señal

- La función del *filtrado de señal* es acondicionar el ruido de las señales eléctricas, manteniendo la integridad de los datos transmitidos por los sensores. Los filtros pueden ser pasivos o activos, y en algunos casos se utilizan algoritmos digitales para este propósito (HBM) (Electricity - Magnetism).

- La *conversión de señal* convierten señales analógicas en digitales y viceversa, lo cual es fundamental ya que muchos sensores en domótica funcionan con señales analógicas, mientras que los sistemas de control operan con señales digitales (HBM) (Electricity - Magnetism).

- El *ajuste de nivel de señal* asegura que la señal esté en el rango adecuado para los dispositivos receptores, amplificando señales débiles o atenuando señales muy fuertes. Esto es importante para evitar la saturación de los sistemas de control (HBM) (Electricity - Magnetism).

- El *aislamiento galvánico* ofrece aislamiento galvánico para evitar interferencias eléctricas y proteger contra sobretensiones, lo cual es vital para la seguridad y fiabilidad de los sistemas domóticos. Este aislamiento también ayuda a evitar corrientes de compensación en lazos de toma de tierra (Pepperl+Fuchs) [13].

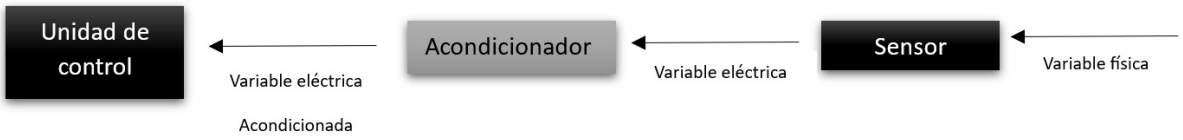


Figura 2.6: Acondicionadores de señal.

2.3. Inteligencia artificial

La inteligencia artificial está desempeñando un papel creciente en la domótica. Los algoritmos de IA pueden aprender de los hábitos y preferencias del usuario para adaptar y optimizar la automatización en el hogar. Esto incluye la capacidad de anticipar las necesidades del usuario, optimizar el consumo de energía y brindar recomendaciones personalizadas [17] .

Algunos ejemplos del uso de la inteligencia artificial son:

1. Visión artificial:

   Se encarga de manejar a los sistemas informáticos, de la capacidad de percibir, procesar y comprender las imágenes y los videos. Algunos ejemplos de aplicaciones de la visión artificial en la domótica son:

   - Reconocimiento facial: Pueden reconocer y verificar a los residentes de una casa através del reconocimiento facial. Esto permite desbloquear puertas, activar perfiles personalizados, ajustar la iluminación y la temperatura de acuerdo con las preferencias de cada individuo.
   - Detección de intrusos: Las cámaras con visión artificial pueden detectar actividad sospechosa en áreas restringidas y enviar notificaciones a los propietarios.
   - Seguimiento de objetos: Los sistemas de visión artificial pueden realizar un seguimiento de objetos en movimiento, como mascotas, para monitorear su actividad y notificar a los propietarios si hay algún comportamiento inusual.
   - Gestión de energía basada en la ocupación: Se puede detectar la presencia de personas en una habitación y ajustar automáticamente la iluminación y la temperatura y así ahorrar energía al minimizar el uso innecesario de dispositivos cuando no hay personas presentes.
   - Detección de incendios: Se pueden detectar el humo o el fuego en una habitación y activar automaticamente las alarmas de incendio, cerrar puertas y ventanas, y alertar a los servicios de emergencia.

2. Reconocimiento de gestos: En la domótica permite que los usuarios controlen y gestionen diversos aspectos de su hogar mediante gestos realizados con las manos u otras partes del cuerpo. Esta tecnología utiliza cámaras o sensores de movimiento para detectar y reconocer los gestos, y luego los traduce en comandos que son interpretados por los dispositivos de domótica.

3. Machine learning: El aprendizaje automático consiste en entrenar modelos de IA para que puedan aprender y mejorar automáticamente a partir de datos. El machine learning en la domótica se utiliza para analizar y comprender los datos generados por los dispositivos y sensores del hogar, permite tomar decisiones y acciones eficientes para mejorar la experiencias del usuario [18].

4. Deep learning: El aprendizaje profundo utiliza redes neuronales artificiales con multiples capas ocultas para procesar y analizar datos complejos. El Deep learning en la domótica se utiliza para resolver problemas más complejos y extraer información valiosa a partir de grandes cantidades de datos generados por los dispositivos y sensores en el hogar [19].

Los sistemas domóticos que se han instalado en la actualidad son aquellos que permiten el control y la automatización de diversos elementos del hogar ya antes mencionados y clasificados según el tipo de conexión arquitectura y el protocolo que utilizan.

2.3.1. Aprendizaje automático

El aprendizaje automático es una rama de la inteligencia artificial que busca brindar a una computadora la capacidad de predecir eventos con base en observaciones del pasado.

Cuando se crea programa de aprendizaje automático, se busca dar a una máquina la habilidad de aprender sin programar explícitamente lo que se busca aprender. Esta es una ciencia que se ha desarrollado algoritmos que se optimizan y descubren relaciones y patrones por sí mismos a manera de hacer predicciones. Cualquier diseño de un modelo que incluya inteligencia artificial cuenta con dos procesos principales: **Entrenamiento e inferencia**. En el entrenamiento se brinda información pasada a un modelo para que la utilice para mejorarse a sí mismo buscando realizar las predicciones esperadas. En el proceso de inferencia se utiliza el modelo creado para hacer predicciones basadas en datos reales que se utilizan como entrada al modelo [20].

2.3.2. Aprendizaje profundo

El aprendizaje profundo es uno de los enfoques más populares para abordar el aprendizaje automático. Este enfoque consiste en crear modelos que son redes que simulan las conexiones de las neuronas en el cerebro humano. El objeto fundamental que simula una neurona es llamado perceptrón. Estos objetos, formar redes, permiten modelar relaciones entre multiples entradas y salidas.

Este es un enfoque flexible pues se puede orientar a resolver diferentes tareas, además de ser una herramienta poderosa para resolver problemas que son adecuados para microcotroladores. Su función principal es extraer patrones de conjuntos de datos mediante la utilización de redes neuronales [21].

Este enfoque utiliza el siguiente flujo de trabajo:

1. Decidir un objetivo.
2. Recolectar un conjunto de datos.
3. Diseñar una arquitectura para el modelo.
4. Entrenar el modelo.

5. Hacer inferencias con el modelo.

6. Evaluar y solucionar problemáticas del modelo.

2.3.3. Aprendizaje automático TinyML

Tiny Machine Learning es una nueva tendencia que busca implementar el aprendizaje automático desde el enfoque de aprendizaje profundo en dispositivos con capacidades limitadas, tal como los microcontroladores. En esencia, se busca utilizar el flujo de trabajo del aprendizaje profundo para desarrollar modelos que sean implementables en sistemas embebidos a través de microcontroladores. Es importante tomar en cuenta que muchos dispositivos de la gama de los microcontroladores tienen limitantes tales como una memoria con poco espacio de almacenaje, capacidad de procesamiento limitada y bajo consumo de potencia, por lo que es necesario modificar los algoritmos comunes de aprendizaje automático para poder utilizarlos de esta forma.

Originalmente, dado que los dispositivos computacionales no contaban con capacidades de procesamiento tan amplias y los algoritmos de aprendizaje automático eran muy complejos y demandaban muchos recursos, era difícil su implementación. Actualmente, con el crecimiento tecnológico y el paso del tiempo, se ha llegado a una época de auge del aprendizaje automático. Esta nueva tendencia brinda la oportunidad de tener acceso a soluciones inteligentes en dispositivos con capacidades limitadas, utilizando el flujo de trabajo del aprendizaje automático como base, aunque implementando ciertas modificaciones para tomar en cuenta las limitantes de los equipos. Este objetivo se ve facilitado ya que el crecimiento de las tecnologías en términos de procesamiento de información y los avances en la optimización de la utilización de recursos en los modelos de aprendizaje automático acercan los campos de la inteligencia artificial y los sistemas embebidos de forma conveniente [22].

2.3.4. Redes neuronales artificiales

Las redes neuronales son capaces de aprender mediante un proceso llamado aprendizaje profundo o deep learning, que son herramientas poderosas para el reconocimiento de gestos y movimientos. Estas tecnologías permiten que los sistemas interpreten y respondan a los movimientos corporales de manera precisa y eficiente.

El sistema propuesto implica identificar y entender los movimientos del cuerpo humano para interpretar o controlar diversas funciones domóticas. Para detectar estos movimientos utilizando sensores como los presentes en la tarjeta Nicla Sense ME, se pueden emplear varios tipos de redes neuronales artificiales (ANNs) adecuadas para procesar señales temporales y secuenciales. A continuación, se describen algunos de estos tipos:

1. Redes neuronales convulucionales (CNN): Las CNNs son efectivas para procesar datos espaciales, como imágenes, y pueden adaptarse para analizar series temporales mediante convoluciones unidimensionales (1D) [23].

Aplicaciones: Extraen características espaciales y temporales de los datos del sensor, ideales para patrones en señales de acelerómetros y giroscopios.

2. Redes Neuronales Recurrentes (RNN): Las RNNs son ideales para datos secuenciales debido a su capacidad de mantener información temporal en sus unidades de memoria internas [24].

   Aplicaciones: Capturan la dinámica temporal de las señales del sensor y son efectivas para modelar secuencias de datos con dependencias temporales.

3. Long Short-Term Memory (LSTM):

   Las LSTM, una variante de las RNN, están diseñadas para superar el problema del gradiente desvanecido, permitiendo capturar dependencias a largo plazo en los datos secuenciales [25]

   Aplicaciones: Analizan secuencias largas de datos de sensores y aprender patrones complejos a lo largo del tiempo.

4. Redes Neuronales de Convolución Temporal (TCN):

   Las TCN combinan las ventajas de las CNN y las RNN para el procesamiento de series temporales, utilizando convoluciones dilatadas para captar patrones en varias escalas temporales [26].

   Aplicaciones: Modelan secuencias temporales largas de manera efectiva sin perder información de la serie.

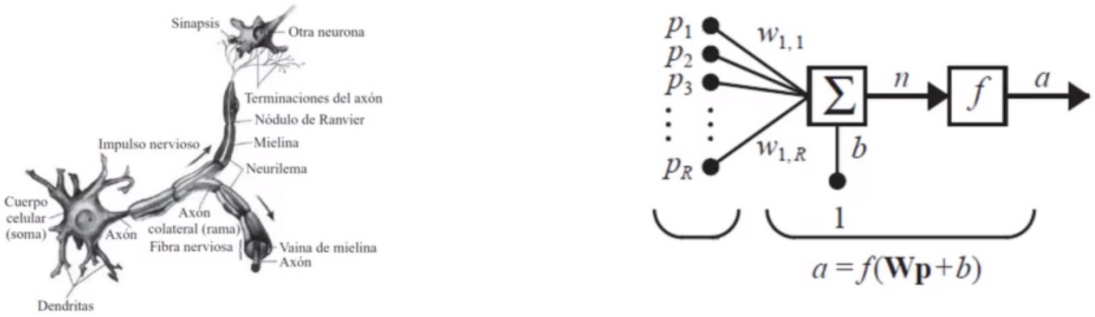


Figura 2.7: Red neuronal natural (izquierda), artificial (derecha)[27].

2.3.5. Modelo simplificado de una red neuronal artificial

Un modelo simplificado de una red neuronal artificial (ANN) se compone de varias capas y elementos clave que permiten procesar y transformar los datos para realizar

predicciones precisas. Las partes fundamentales de una ANN y su funcionamiento básico son:

Capa de entrada

La capa de entrada recibe la información inicial que puede ser de cualquier tipo, como imágenes, texto o valores numéricos. Esta capa no realiza ningún procesamiento, solo pasa los datos a las siguientes capas [28].

Capas Ocultas:

Las capas ocultas estan compuestas por múltiples neuronas artificiales que procesan y transforman los datos de entrada. Cada neurona en una capa oculta toma las conexiones de la capa anterior y aplica una función de activación para extraer características relevantes de los datos [29].

Capa de Salida:

La capa de salida genera la respuesta final o predicción de la red neuronal. El número de neuronas en esta capa depende del tipo de problema a resolver. Por ejemplo, en un problema de clasificación binaria, habrá una neurona que produce un valor entre 0 y 1, indicando la probabilidad de cada clase [30].

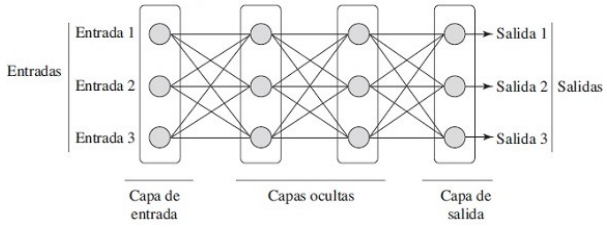


Figura 2.8: Partes fundamentales de una ANN [31]

Conexiones Ponderadas:

Las conexiones entre las neuronas tienen pesos asociados que determinan la importancia de cada conexión. Durante el entrenamiento, estos pesos se ajustan para minimizar el error de predicción y mejorar el rendimiento de la red [32].

Función de Coste:

La función de coste evalúa la discrepancia entre la salida de la red y la salida deseada. El objetivo del entrenamiento es minimizar esta función de coste ajustando los pesos de las conexiones [33].

Algoritmo de Optimización:

Un algoritmo de optimización, como el descenso de gradiente, se utiliza durante el entrenamiento para ajustar los pesos de las conexiones y minimizar la función de coste. Este proceso mejora la precisión de las predicciones de la red [34].

2.4. Componentes de una red neuronal artificial

La red neuronal artificial se compone por:

1. Entradas (P): Las entradas son los datos que recibe la neurona. Estos datos pueden ser cualquier tipo de información que se desea procesar o analizar mediante la red neuronal [35]. En la Figura 2.9 se muestra la representación de una entrada.

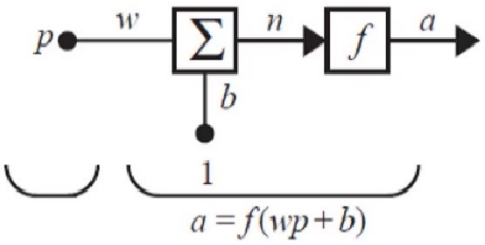


Figura 2.9: Entradas a la red neuronal.

2. Pesos (W): Los pesos son coeficientes que se asignan a cada entrada de la neurona. Estos coeficientes ponderan la importancia de cada entrada en el proceso de activación de la neurona. En la Figura 2.10 se muestra una imagen de los pesos [35].

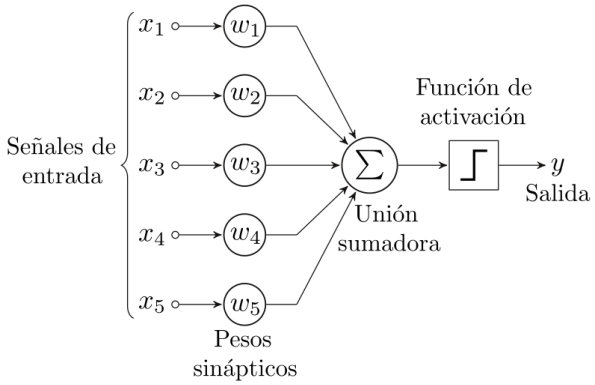


Figura 2.10: Pesos W (coeficientes).

3. Bías (B): Es un término adicional que se suma a la entrada ponderada de la neurona antes de aplicar la función de activación.En la Figura 2.11 se muestra una imagen de la bías.

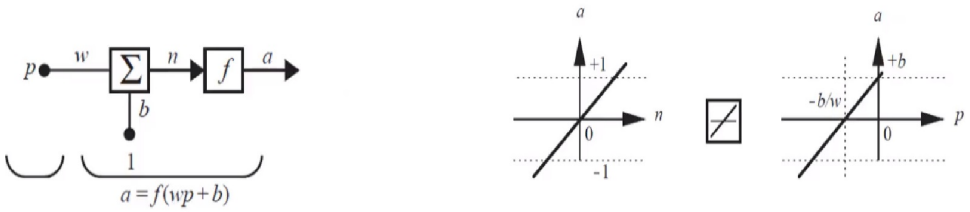


Figura 2.11: Bias B.

4. Función de activación o transferencia (F): Las funciones de activación se utilizan para introducir la no linealidad en las redes neuronales. En la Figura 2.12 se muestra una representación de la transferencia.

Figura 2.12: Transferencia F.

2.5. Estado del arte

El reconocimiento de movimientos y la emisión de alertas se ha convertido en un área de interés creciente debido a sus aplicaciones en salud, seguridad y automatización. Los avances en sensores, algoritmos de aprendizaje automático y sistemas inteligentes han permitido el desarrollo de soluciones más precisas y eficientes para la detección de movimientos específicos y la emisión de alertas en tiempo real:

Sensores Portátiles

Estos dispositivos utilizan acelerómetros y giroscopios para calcular pasos y reconocer movimientos. Aunque son prácticos, pueden presentar errores en la identificación precisa de movimientos complejos.

Uso de dispositivos inteligentes como relojes o pulseras para detección de actividad física.

Los relojes o las pulseras utilizan acelerómetros y giroscopios para rastrear la actividad física diaria, calcular pasos y reconocer diferentes tipos de movimientos [36].

Figura 2.13: Relojes inteligentes.

Sensores de tejido de punto

Utilizados ampliamente en el campo de la salud deportiva, estos sensores aprovechan la deformación por tensión del tejido para reflejar movimientos del cuerpo humano, cambiando la señal eléctrica. Tienen alta sensibilidad y estabilidad durante el movimiento, y permiten distinguir con precisión las etapas del ciclo de la marcha.

Sensores de pantalones deportivos de tejido de punto utilizados para monitorear el ciclo de la marcha.

Estos sensores, integrados en ropa deportiva, detectan movimientos a través de cambios en la resistencia eléctrica del tejido. Son altamente elásticos y estables, permitiendo una monitorización precisa del ciclo de la marcha en etapas de postura y balanceo[37].

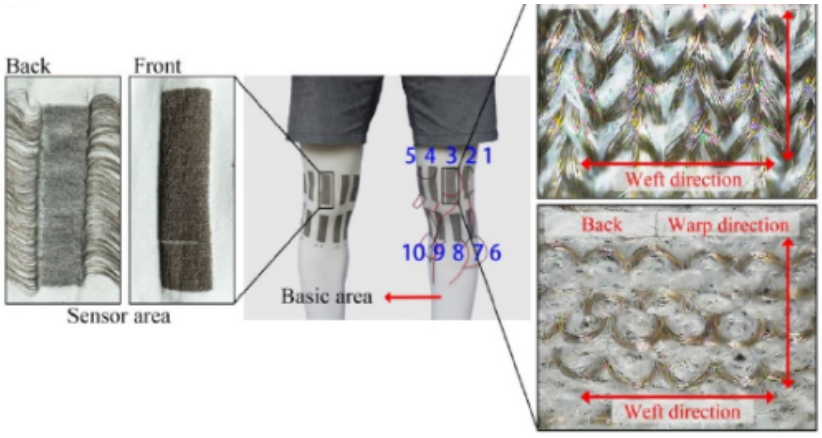


Figura 2.14: Sensores de Tejido de Punto [37].

Algoritmos basados en datos de imágenes y videos

Utilización de redes convolucionales 3D y modelos inflados (I3D) para analizar secuencias de video y detectar comportamientos anómalos, aplicados en vigilancia de tráfico y seguridad pública.

Cámaras de vigilancia en el reconocimiento de acciones humanas y la detección de accidentes en sistemas de seguridad pública.

Utilizan redes convolucionales 3D y modelos inflados (I3D) para analizar secuencias de video, detectando comportamientos anómalos y mejorando la seguridad pública mediante la vigilancia automatizada [38].

Figura 2.15: Camaras de reconocimiento de acciones humanas [39].

Monitoreo de salud

Sistemas que detectan caídas y movimientos anómalos en personas mayores, emitiendo alertas a cuidadores. Estos sistemas utilizan sensores portátiles y algoritmos de aprendizaje automático para identificar caídas y enviar alertas de emergencia.

Sistemas de detección de caídas para personas mayores, como life alert.

Utilizan sensores portátiles y algoritmos de aprendizaje automático para detectar caídas y movimientos anómalos, enviando alertas inmediatas a cuidadores o servicios de emergencia [40].

Figura 2.16: Camaras de reconocimiento de acciones humanas [41].

Seguridad en el trabajo

Sensores integrados en equipos de protección personal que detectan movimientos peligrosos y emiten alertas inmediatas, ayudando a prevenir accidentes laborales.

Sensores integrados en equipos de protección personal utilizados en fábricas.

Detectan movimientos peligrosos y emiten alertas inmediatas para prevenir accidentes laborales. Estos sensores pueden estar integrados en cascos, chalecos y otros equipos de seguridad [42].

Figura 2.17: Sensores integrados en equipos de protección personal utilizados en fábricas [43].

Un sistema de reconocimiento de movimientos utilizado en la domótica puede tener muchas ventajas, como mejorar la comodidad, la accesibilidad sobre todo para perso-

nas que tienen alguna discapacidad o movimientos limitados, la seguridad y el ahorro energético de las personas. Sin embargo, también puede tener algunos desafíos y limitaciones, como la variabilidad de los gestos entre las personas, las condiciones ambientales, la precisión, la velocidad, la privacidad y sobre todo el costo que representa el tener un sistema con estas características.

Además de los diferentes productos y servicios ya antes mencionados que pueden ofrecer las empresas sobre las distintas formas de como controlar un sistema domótico, esta investigación tiene como objetivo realizar un sistema de reconocimiento de gestos para controlar un sistema domótico y poder solucionar algunos obstáculos que hay en la actualidad.

Capítulo 3

Componentes básicos para el sistema

En este capítulo se incluyen algunas definiciones y elementos que forman parte del sistema de reconocimiento de movimiento, junto con el aprendizaje automático para controlar alguna gestión de la domótica. Esto incluye el tipo de sensor, los actuadores, tarjetas de desarrollo o microcontroladores, así como el software para la configuración, parametrización o visualización.

3.1. Gestos y movimientos corporales

Antes de explicar el diseño y desarrollo del sistema de reconocimiento de movimientos, es fundamental establecer con claridad qué tipo de gestos o movimientos se utilizarán. Es bien conocido que la efectividad de cualquier sistema depende en gran medida de la comprensión detallada y precisa de los movimientos que se pretende reconocer e interpretar.

Es relevante destacar que el lenguaje corporal, componente esencial en el reconocimiento de gestos, se refiere al conjunto de señales no verbales que las personas emplean para comunicar sus pensamientos, emociones y actitudes. A diferencia del lenguaje verbal, que se expresa mediante palabras, el lenguaje corporal engloba una variedad de elementos, como gestos específicos, expresiones faciales, posturas y otros movimientos del cuerpo. Esta forma de comunicación añade capas adicionales de significado y contexto a la interacción humana. Para lograrlo, comenzaremos con algunas definiciones.

3.1.1. Movimientos

Los movimientos se refieren a cambios o desplazamientos físicos que realiza un objeto, una parte del cuerpo o una entidad en el espacio. Estos movimientos pueden ser voluntarios o involuntarios y están presentes en una amplia gama de contextos, desde el movimiento de las personas y los animales hasta el movimiento de objetos inanimados.

En el ámbito biológico y humano, los movimientos pueden ser acciones físicas ejecutadas por los músculos del cuerpo, como caminar, correr, saltar, agarrar objetos, entre

otros. Estos movimientos pueden ser controlados conscientemente por el individuo o pueden ser respuestas automáticas del cuerpo a estímulos externos.

3.1.2. Tipos de movimientos

Existen varios tipos de movimientos, que pueden clasificarse de diversas formas según su naturaleza y función. Aquí hay una descripción de algunos tipos comunes de movimientos:

- *Movimientos voluntarios:* Estos movimientos son controlados conscientemente por el individuo y se realizan de manera intencional. Ejemplos incluyen caminar, correr, saltar, agarrar objetos, escribir y hablar.
- *Movimientos involuntarios:* Estos movimientos ocurren sin que la persona tenga control consciente sobre ellos y son controlados por el sistema nervioso autónomo. Ejemplos incluyen el latido del corazón, la respiración, el parpadeo y los movimientos peristálticos del sistema digestivo.
- *Movimientos reflejos:* Son respuestas automáticas del cuerpo a estímulos externos, como el reflejo de estiramiento, el reflejo de retirada y el reflejo de la tos. Estos movimientos son controlados principalmente por la médula espinal y el tronco cerebral.
- *Movimientos de locomoción:* Se refieren al desplazamiento del cuerpo de un lugar a otro y pueden incluir caminar, correr, gatear, nadar y volar, dependiendo de la especie y las capacidades físicas del organismo.
- *Movimientos articulares:* Estos movimientos involucran el movimiento de las articulaciones del cuerpo y pueden clasificarse en diferentes tipos, como flexión, extensión, abducción, aducción, rotación, pronación y supinación.

3.1.3. Gestos

Los gestos son movimientos corporales específicos que realizamos con partes de nuestro cuerpo, como manos, brazos, cabeza, cara u otras partes, para comunicar mensajes o expresar emociones. Los gestos son una forma importante de lenguaje no verbal y pueden ser utilizados consciente o inconscientemente para transmitir información en diversas situaciones. Aquí hay algunos tipos comunes de gestos:

3.1.4. Tipos de gestos

- **Gestos de manos:** Incluyen movimientos de las manos y los dedos. Algunos ejemplos son levantar el pulgar para expresar aprobación, hacer un gesto de ok con los dedos o señalar con el dedo índice para indicar algo específico.

- **Gestos faciales:** Involucran expresiones faciales que comunican emociones o intenciones. Por ejemplo, sonreír para mostrar felicidad, fruncir el ceño para expresar preocupación o levantar las cejas en señal de sorpresa.
- **Gestos de cabeza:** Movimientos de la cabeza que pueden tener diversos significados. Asentir con la cabeza puede indicar acuerdo o comprensión, mientras que negar con la cabeza puede expresar desacuerdo o rechazo.
- **Gestos corporales:** Incluyen movimientos de todo el cuerpo. Una postura relajada y abierta puede sugerir confianza, mientras que cruzar los brazos puede indicar defensividad o desacuerdo.
- **Gestos de ademanes:** Movimientos expresivos de los brazos y las manos que acompañan o refuerzan el habla. Por ejemplo, gesticular con las manos mientras se explica algo o dar palmadas para enfatizar un punto.
- **Gestos emblemáticos:** Gestos que tienen un significado cultural específico y a menudo se utilizan en lugar de palabras. Un ejemplo común es el pulgar levantado como señal de "bien.º "buen trabajo".
- **Gestos ilustrativos:** Acompañan y complementan el habla, ayudando a visualizar o enfatizar las palabras habladas. Por ejemplo, hacer un círculo con las manos para representar algo completo o señalar hacia adelante para indicar dirección.

3.2. Elección de movimientos

Flexión y extensión del brazo y la mano

Para nuestra tesis, hemos elegido enfocarnos en los movimientos de flexión y extensión del brazo y la mano debido a varias razones fundamentales:

- **Simplicidad y claridad:** La flexión y extensión son movimientos básicos y fácilmente identificables, lo que reduce la complejidad en la fase de reconocimiento y mejora la precisión del sistema.
- **Relevancia funcional:** Estos movimientos son fundamentales en muchas actividades diarias y laborales. Monitorearlos puede proporcionar información crucial para la rehabilitación física y la ergonomía.
- **Aplicabilidad:** La detección precisa de estos movimientos puede aplicarse en diversas áreas como la asistencia a personas con movilidad reducida, la automatización del hogar y la seguridad laboral.
- **Simplicidad en el modelado:** Al ser movimientos relativamente simples, los modelos de aprendizaje automático pueden ser entrenados de manera más efectiva y rápida, utilizando menos recursos computacionales.

Estos factores hacen que la elección de los movimientos de flexión y extensión del brazo y la mano sea ideal para el desarrollo de un sistema de reconocimiento de movimiento preciso y eficiente, que puede ser aplicado en múltiples contextos para mejorar la calidad de vida y la seguridad de los usuarios.

3.3. Reconocimiento de movimientos

El reconocimiento de movimientos en el ámbito del aprendizaje automático (Machine Learning) es un proceso que implica identificar y clasificar patrones de movimiento a partir de datos de entrada, como señales de sensores, imágenes o videos. Este proceso tiene aplicaciones en una variedad de campos, incluyendo la salud, el fitness, la seguridad, la robótica, los videojuegos y la realidad virtual, entre otros.

El reconocimiento de movimientos utiliza algoritmos de aprendizaje automático para analizar y procesar datos de entrada con el fin de reconocer y entender los movimientos realizados por un individuo, un objeto o un sistema. Estos algoritmos pueden entrenarse utilizando conjuntos de datos etiquetados que contienen ejemplos de diferentes tipos de movimientos, lo que permite al sistema aprender a distinguir entre ellos y realizar predicciones precisas sobre el tipo de movimiento presente en nuevos datos de entrada.

3.3.1. Gestiones de la domótica que pueden ser controlados mediante movimientos

Como se mencionó en el Capítulo 1, hay tres sistemas a gestionar en la domótica:

Gestión de la energía

La gestión de energía se encarga de gestionar el consumo de energía mediante temporizadores, relojes programadores, termostatos, etc.

Tabla 3.1: Gestión de Energía.

Sistema a gestionar	**Funciones**
Calefacción/Climatización	Programación y zonificación de la climatización y equipos domésticos.
Racionalización de cargas	Desconexión de equipos de uso no prioritario en función del consumo eléctrico en un momento dado.
Encendido y apagado	Apagado general de todas las luces de la vivienda y automatización del apagado/encendido de cada punto de luz.

Gestión del confort

La gestión del confort y la calidad de vida nos proporciona una serie de comodidades, especialmente para personas mayores o con problemas de movilidad, cognitivos o con algún tipo de minusvalía.

Tabla 3.2: Gestión de Confort.

Sistema a gestionar	**Funciones**
Comunicación a distancia	Integración del video timbre, chapas inteligentes, cámaras, la conexión de tus dispositivos y electrodomésticos.
Racionalización de cargas	Desconexión de equipos de uso no prioritario en función del consumo eléctrico en un momento dado.
Iluminación	Regulación automática de la iluminación según el nivel de luminosidad del ambiente.
Control de persianas y riego	Accionamiento automático de persianas y toldos, y control de sistema de riego.
Conexión y automatización de puertas y ventanas	Comunicación directa con el exterior y contar con un apoyo de acuerdo a las necesidades del usuario.

Gestión de la seguridad

La gestión de la seguridad y vigilancia que proporciona un sistema domótico es más amplia que la que nos puede proporcionar cualquier otro sistema.

Tabla 3.3: Gestión de Seguridad.

Sistema a gestionar	**Funciones**
Detección de movimiento	Control de presencia y detección de intrusismo.
Detección de robo	Detección de rotura de cristales y forzado de puertas.
Detección de presencia	Gestión del control de acceso con reconocimiento o identificación de los usuarios.

3.3.2. La captación de movimientos

La captación de movimientos para crear un modelo de Machine Learning es un proceso crucial en el desarrollo de sistemas que pueden reconocer y comprender patrones de movimiento a partir de datos de entrada. Este proceso implica la recopilación y el registro de datos de movimiento utilizando diversos sensores, como acelerómetros, giroscopios, cámaras de video u otros dispositivos de captura de movimiento.

El objetivo de la captación de movimientos es recopilar datos de calidad que representen con precisión los diferentes tipos de movimientos que se desean reconocer y clasificar. Esto puede implicar la realización de diversas actividades o gestos por parte de individuos en entornos controlados, mientras se registran las señales de movimiento con los sensores adecuados.

Una vez que se han recopilado los datos, se pueden utilizar técnicas de preprocesamiento para limpiar, filtrar y normalizar los datos antes de alimentarlos a un modelo de Machine Learning. Esto puede incluir la eliminación de ruido, la interpolación de datos faltantes, la normalización de las escalas de los datos y la segmentación de los datos en ventanas temporales para su análisis.

El siguiente paso es entrenar un modelo de Machine Learning utilizando los datos de movimiento recopilados. Esto implica alimentar los datos al modelo junto con las etiquetas correspondientes que indican el tipo de movimiento realizado. Dependiendo del tipo de problema y los datos disponibles, se pueden utilizar diferentes tipos de modelos de Machine Learning, como clasificadores lineales, redes neuronales, árboles de decisión o métodos de aprendizaje profundo.

Una vez que el modelo ha sido entrenado y validado, puede ser desplegado en un sistema en tiempo real donde puede reconocer y clasificar automáticamente los movimientos capturados por los sensores. Esto puede tener aplicaciones en una variedad de campos, como la salud, el fitness, la seguridad, la realidad virtual, los videojuegos y la interacción humano-computadora, entre otros.

3.3.3. Sensores (giroscopios y acelerómetros)

Los sensores de giroscopios y acelerómetros son componentes clave en los sistemas de reconocimiento de movimientos, ya que proporcionan datos cruciales sobre la orientación y el movimiento de un objeto en el espacio. Se pueden clasificar como:

- *Acelerómetros:* Los acelerómetros miden la aceleración lineal experimentada por un objeto en una o más direcciones. Estos sensores pueden detectar cambios en la velocidad y la dirección del movimiento, lo que permite determinar la orientación, la posición y los cambios de movimiento de un objeto. En el contexto del reconocimiento de movimientos, los acelerómetros pueden utilizarse para detectar gestos, pasos, cambios de dirección y otras acciones físicas mediante la medición de la aceleración experimentada por el dispositivo en diferentes ejes.

- *Giroscopios:* Los giroscopios miden la velocidad angular o la tasa de cambio de la orientación de un objeto en el espacio. Estos sensores proporcionan información sobre la velocidad de rotación y la dirección del movimiento de un objeto, lo que permite determinar la orientación y los cambios de dirección en tiempo real. En el reconocimiento de movimientos, los giroscopios pueden utilizarse para detectar giros, inclinaciones y otras acciones que implican cambios en la orientación del dispositivo.

A continuación, se presentan tres tipos de sensores que pueden ser utilizados en esta tesis, junto con sus descripciones:

Sensor	Descripción	Aplicaciones
LSM6DS33TR	Giroscopio y acelerómetro de 3 ejes.	Ofrece una combinación de datos de aceleración y velocidad angular con alta precisión y bajo consumo de energía, ideal para aplicaciones en dispositivos portátiles y seguimiento de movimientos.
MPU6050	Unidad de medida inercial que combina un acelerómetro de 3 ejes y un giroscopio de 3 ejes.	Proporciona datos de aceleración y rotación con un procesador de movimiento digital que puede fusionar los datos del sensor para obtener una orientación precisa. Utilizado en aplicaciones como drones, dispositivos portátiles y controladores de juegos.
Nicla Sense ME	Tarjeta de desarrollo que incluye múltiples sensores como acelerómetro, giroscopio, sensor de presión, y sensor de gas.	Ideal para proyectos de IoT y monitoreo ambiental, permite capturar y procesar una amplia gama de datos sensoriales. La integración con la plataforma Arduino facilita el desarrollo rápido de aplicaciones de detección de movimiento y análisis de datos.

Tabla 3.4: Descripción de los sensores LSM6DS33TR, MPU6050 y Nicla Sense ME.

3.3.4. Captación de movimientos con la Nicla Sense ME

La captación de movimientos utilizando la Nicla Sense ME es un proceso esencial para desarrollar un modelo de Machine Learning (ML) capaz de reconocer patrones de movimiento específicos. Este proceso se realiza a través de los sensores integrados de alta precisión en la Nicla Sense ME, los cuales permiten registrar datos detallados sobre

movimientos del brazo y la mano. A continuación, se describen los pasos generales del proceso:

1. **Configuración inicial:** Conectar la Nicla Sense ME a una computadora e instalar el entorno de desarrollo Arduino IDE. Asegurarse de tener instaladas las bibliotecas necesarias para la Nicla Sense ME y los sensores incluidos.

2. **Programación:** Escribir un código en Arduino IDE que configure los sensores de acelerómetro y giroscopio para capturar datos de movimiento. El código debe ser capaz de leer los datos de estos sensores y almacenarlos en una variable o enviar los datos a través de una conexión serial para su análisis posterior.

3. **Captura de Datos:** Con la Nicla Sense ME correctamente programada y conectada, realizar los movimientos de flexión y extensión del brazo y la mano. Asegurarse de realizar estos movimientos en un entorno controlado para obtener datos precisos y consistentes.

4. **Almacenamiento de datos:** Los datos capturados deben ser convertidos a un formato CSV para su posterior análisis. Estos datos pueden ser almacenados en una tarjeta SD conectada a la Nicla Sense ME o enviados directamente a una computadora. Asegurarse de que los datos estén correctamente etiquetados con el tipo de movimiento correspondiente.

5. **Preprocesamiento:** Utilizar técnicas de preprocesamiento para limpiar y normalizar los datos capturados. Esto puede incluir la eliminación de ruido, la interpolación de datos faltantes y la segmentación de los datos en ventanas temporales.

6. **Entrenamiento del modelo:** Usar los datos preprocesados para entrenar un modelo de Machine Learning en la plataforma Neuton Tiny ML. Esto implica cargar el archivo CSV en la plataforma, alimentar los datos al modelo junto con las etiquetas correspondientes que indican el tipo de movimiento realizado y utilizar técnicas de validación cruzada para asegurarse de que el modelo generaliza bien a nuevos datos.

7. **Implementación:** Una vez que el modelo ha sido entrenado y validado, puede implementarse en un sistema en tiempo real para reconocer y clasificar automáticamente los movimientos capturados por los sensores de la Nicla Sense ME.

3.4. Apredizaje automático TinyML

El aprendizaje automático TinyML se refiere a la implementación de modelos de aprendizaje automático en dispositivos de baja potencia y recursos limitados, como microcontroladores, sistemas embebidos y dispositivos IoT (Internet de las cosas). Estos modelos están optimizados para funcionar eficientemente en hardware con restricciones de memoria, energía y procesamiento.

3.4.1. TinyML y como se diferencia del aprendizaje automático tradicional

TinyML, o aprendizaje automático en dispositivos de baja potencia, se diferencia del aprendizaje automático tradicional en varios aspectos clave:

- *Recursos limitados*

  En TinyML, los modelos de aprendizaje automático están diseñados para ejecutarse en dispositivos con recursos limitados, como microcontroladores y sistemas embebidos, que tienen restricciones de memoria, energía y procesamiento.

- *Eficiencia energética*

  La eficiencia energética es una consideración crítica en TinyML, ya que los dispositivos de baja potencia a menudo funcionan con baterías o energía limitada. Los modelos de TinyML están optimizados para minimizar el consumo de energía y prolongar la vida útil de la batería, lo que puede implicar técnicas como la cuantización de parámetros, la poda de redes neuronales y la ejecución en hardware especializado de bajo consumo.

- *Tamaño del modelo*

  En TinyML, los modelos deben ser lo suficientemente pequeños como para caber en la memoria limitada de los dispositivos de baja potencia. Esto requiere técnicas de compresión y optimización para reducir el tamaño del modelo sin sacrificar demasiada precisión. Los modelos de TinyML suelen ser mucho más pequeños que los modelos de aprendizaje automático tradicionales, lo que los hace más adecuados para dispositivos con recursos limitados.

- *Entrenamiento en el borde*

  En TinyML, el entrenamiento del modelo a menudo se realiza en el borde, es decir, en el propio dispositivo o en un dispositivo cercano, en lugar de en servidores remotos en la nube. Esto permite que los modelos se adapten y mejoren continuamente en función de los datos locales, lo que puede ser beneficioso en entornos donde la conectividad a Internet es limitada o poco confiable.

- *Aplicaciones específicas*

  TinyML se utiliza principalmente en aplicaciones que requieren inferencia en tiempo real en dispositivos de baja potencia, como el reconocimiento de voz, la detección de gestos, la monitorización de la salud, el control de dispositivos IoT y la detección de anomalías en tiempo real. Estas aplicaciones tienen requisitos únicos en términos de rendimiento, precisión y eficiencia energética, que deben abordarse mediante técnicas especializadas de TinyML.

3.4.2. Aplicaciones prácticas de TinyML en la vida cotidiana

- *Dispositivos portátiles de salud y fitness*

  Los dispositivos portátiles, como smartwatches y pulseras de actividad, utilizan TinyML para realizar un seguimiento de la actividad física, monitorizar el ritmo cardíaco, calcular el gasto calórico y detectar patrones de sueño. Esto permite a los usuarios mejorar su salud y bienestar mediante la monitorización continua y el análisis de sus datos fisiológicos.

- *Asistentes de voz y controladores de hogar inteligente*

  Los asistentes de voz, como Google Home y Amazon Echo, así como los controladores de hogar inteligente, utilizan TinyML para reconocer y comprender comandos de voz, como encender luces, ajustar la temperatura y reproducir música. Esto permite a los usuarios controlar dispositivos y realizar tareas en el hogar de forma manos libres y conveniente.

- *Dispositivos de seguridad y vigilancia*

  Los sistemas de seguridad y vigilancia, como cámaras de seguridad y alarmas, utilizan TinyML para detectar y reconocer actividades sospechosas, como intrusiones, robos y movimientos inusuales. Esto ayuda a proteger hogares y negocios mediante la detección temprana de amenazas y la notificación rápida a los propietarios o autoridades correspondientes.

- *Dispositivos de automoción y transporte*

  Los sistemas de automoción y transporte, como vehículos autónomos, drones y robots de entrega, utilizan TinyML para detectar y evitar obstáculos, reconocer señales de tráfico y peatones, y tomar decisiones en tiempo real basadas en el entorno circundante. Esto permite una conducción más segura y eficiente, así como una entrega de paquetes más rápida y precisa.

- *Dispositivos médicos y de asistencia*

  Los dispositivos médicos y de asistencia, como prótesis, dispositivos de rehabilitación y sistemas de monitorización de pacientes, utilizan TinyML para adaptarse a las necesidades individuales de los usuarios, detectar cambios en la salud y proporcionar asistencia personalizada. Esto mejora la calidad de vida de las personas con discapacidades y enfermedades crónicas al ofrecerles mayor autonomía y apoyo.

3.4.3. Problemas más comunes al desarrollar modelos de TinyML

Desarrollar modelos de TinyML puede enfrentar retos debido a las limitaciones de memoria, capacidad de procesamiento y energía. La optimización del modelo para ajustarse a estas restricciones es crucial,y aunque se complica requiere de técnicas específicas

de compresión y cuantización. Además, la disponibilidad y calidad de los datos etiquetados pueden ser limitadas, lo que dificulta la recolección y el etiquetado preciso de los datos de entrenamiento. Validar y probar los modelos en entornos reales también puede ser desafiante debido a las variaciones en los datos y las condiciones del mundo real.

3.5. Microcontroladores y procesadores

Los procesadores y microcontroladores son componentes esenciales en los sistemas electrónicos, pero tienen características y aplicaciones diferentes.

Un sistema microprogramable es aquel que, mediante electrónica digital encapsulada en uno o varios circuitos integrados y con un generador de pulsos de alta velocidad, es capaz de seguir una secuencia de instrucciones. Un sistema microprogramable consta de los siguientes bloques:

- **Oscilador o generador de pulsos:** Genera los pulsos necesarios para que el sistema vaya perfectamente sincronizado. Por cada pulso de reloj se ejecutan una o varias instrucciones en el bloque CPU.

- **Unidad central de proceso (CPU):** Se encarga de ejecutar las instrucciones de los programas, así como realizar las operaciones aritmético-lógicas que se requieran durante la ejecución de dichos programas.

- **Unidad de memoria:** Almacena los programas que se van a ejecutar y los resultados derivados de dicha ejecución.

- **Bloque de entrada y salida:** Gobierna el flujo de datos que existe entre el exterior y el interior del sistema. En el exterior contamos con los periféricos, que son dispositivos que introducen información al sistema.

- **Periféricos:** Son dispositivos microprogramables o simplemente circuitos digitales que permiten al usuario interactuar con el sistema.

El microprocesador y el microcontrolador son dos sistemas microprogramables que contienen todos los bloques anteriormente mencionados, pero con algunas diferencias:

Microprocesador

Un microprocesador es un componente central de una computadora que realiza operaciones aritméticas y lógicas, y ejecuta instrucciones de programas almacenadas en la memoria. Es el cerebro principal de una computadora y se utiliza en sistemas de propósito general.

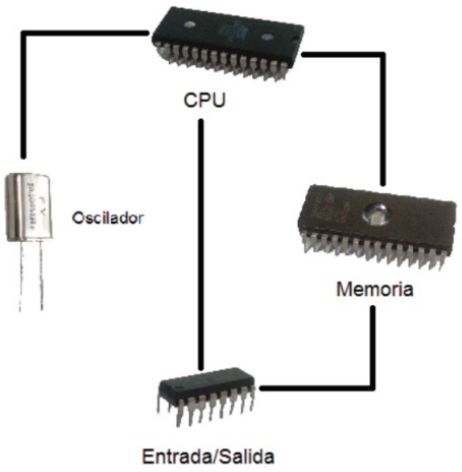


Figura 3.1: Microprocesador.

Microcontrolador

Un microcontrolador es un dispositivo integrado que combina un microprocesador con memoria, periféricos de entrada/salida y, a veces, otros componentes, todo en un solo chip. Está diseñado para controlar operaciones específicas en sistemas embebidos.

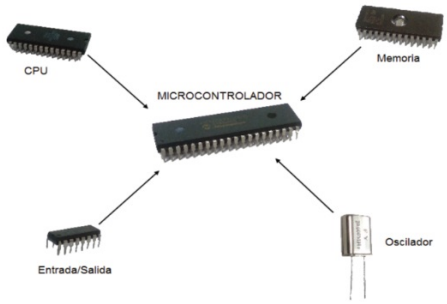


Figura 3.2: Microcontrolador.

3.5.1. Microcontroladores para ML

Los microcontroladores para Machine Learning (ML) son componentes esenciales en sistemas de TinyML, permitiendo la implementación de modelos de aprendizaje automático en dispositivos de baja potencia. Estos microcontroladores están diseñados para ejecutar modelos de ML en el borde de manera eficiente y efectiva. Características clave incluyen eficiencia energética, rendimiento suficiente para ejecutar modelos ML, y limitaciones de memoria y almacenamiento. Además, ofrecen soporte de hardware y software específico para el desarrollo de modelos de ML en el borde, así como capacidades de conectividad para la comunicación con otros dispositivos o la transmisión de datos a la nube.

3.5.2. Microcontroladores para TinyML

La implementación de algoritmos de aprendizaje automático en microcontroladores es un área de oportunidad en constante desarrollo y va ganando relevancia con la creciente demanda de sistemas inteligentes [6]. Aunque los microcontroladores están diseñados para construir sistemas embebidos bajo tareas específicas, estos dispositivos están limitados en términos de capacidad de procesamiento y memoria, lo que requiere una adecuada depuración del código.

A continuación, se presenta una tabla con algunos microcontroladores que se utilizan en el contexto de aprendizaje automático:

Microcontrolador	Descripción	Aplicaciones
Arduino Nano BLE Sense	Contiene sensores como acelerómetro, giroscopio, magnetómetro y micrófono, con conectividad Bluetooth de bajo consumo.	Ideal para aplicaciones móviles e inalámbricas, permitiendo reconocimiento de patrones, detección de problemas y monitoreo de la salud en tiempo real.
Raspberry Pi Pico	Microcontrolador RP2040 de doble núcleo con opciones de entrada y salida variadas.	Adecuado para proyectos de TinyML como reconocimiento de voz, clasificación de imágenes y detección de movimientos, compatible con TensorFlow Lite.
Nicla Sense ME	Integra múltiples sensores como acelerómetro, giroscopio, sensor de presión, humedad, temperatura y calidad del aire.	Ideal para proyectos de IoT y monitoreo ambiental, con capacidad para capturar y procesar datos sensoriales de forma eficiente.

Tabla 3.5: Microcontroladores para aprendizaje automático TinyML.

3.6. Plataformas para programar microcontroladores

La plataforma *Arduino* Arduino es una plataforma de código abierto que simplifica la creación de proyectos electrónicos. Es compatible con una amplia gama de microcontroladores y ofrece un IDE fácil de usar [44].

La plataforma *PlatformIO* es un entorno de desarrollo integrado (IDE) multiplataforma que soporta una gran variedad de microcontroladores y placas de desarrollo. Es

especialmente útil para proyectos de IoT y aplicaciones embebidas [45].

La plataforma *Mbed* es una plataforma de desarrollo en la nube creada por Arm para el desarrollo de aplicaciones embebidas utilizando microcontroladores Arm Cortex-M [46].

La plataforma *Espressif IDF (ESP-IDF)* es el entorno de desarrollo oficial para los microcontroladores ESP32 de Espressif Systems, proporcionando un conjunto de herramientas avanzadas para el desarrollo de aplicaciones IoT [47].

La plataforma *STM32Cube* es un conjunto de herramientas de desarrollo proporcionado por STMicroelectronics, que incluye bibliotecas de software y herramientas de configuración para microcontroladores STM32 [48].

La plataforma *TensorFlow Lite for Microcontrollers* es una versión optimizada del popular framework de aprendizaje automático TensorFlow, diseñada para ejecutarse en dispositivos con recursos limitados como microcontroladores [49].

La plataforma *Edge Impulse* es una plataforma de desarrollo que facilita la creación, implementación y gestión de modelos de aprendizaje automático en dispositivos embebidos, especialmente microcontroladores [50].

Figura 3.3: Plataformas para programar microcontroladores.

3.7. Plataformas para desarrollar modelos ML

El lenguaje de programación *Python*, que es interpretado y de alto nivel, se utiliza ampliamente en múltiples aplicaciones. Se destaca por su simplicidad y legibilidad de código, lo que facilita su uso por los programadores independientemente de su experiencia [51].

Edge Impulse es una plataforma de desarrollo de aprendizaje automático para dispositivos tipo *Edge*. Permite la creación y entrenamiento de modelos de aprendizaje automático y su implementación en una variedad de microcontroladores y placas de desarrollo, incluidas algunas de Arduino y STM32 [50].

Figura 3.4: Edge impulse [50].

El *Espressif IDF* IoT Development Framework (ESP-IDF) es el entorno de desarrollo oficial para los microcontroladores ESP32 de Espressif Systems. Ofrece soporte para TensorFlow Lite Micro y otras herramientas de aprendizaje automático, lo que permite la implementación de modelos en dispositivos ESP32 [47].

Figura 3.5: Espressif [47].

PlatformIO es una plataforma de desarrollo multiplataforma que proporciona soporte para una variedad de microcontroladores y placas, incluidas algunas utilizadas en aplicaciones de aprendizaje automático, como los microcontroladores ESP32 y STM32. Se puede integrar con frameworks de aprendizaje automático como TensorFlow Lite [45].

TensorFlow es una biblioteca de código abierto desarrollada por Google que se utiliza para aplicaciones de aprendizaje automático y en particular para el desarrollo y entrenamiento de modelos de redes neuronales. TensorFlow tiene una amplia gama de aplicaciones, incluyendo reconocimiento de imágenes, procesamiento de lenguaje natural, clasificación de datos, generación de texto y diversas tareas relacionadas con el aprendizaje automático. Es utilizado tanto en la investigación académica como en aplicaciones comerciales, y ha sido adoptado por muchas empresas y organizaciones para construir soluciones basadas en aprendizaje automático [52].

Figura 3.6: TensorFlow [52].

Neuton ML es una plataforma integral para el desarrollo y despliegue de modelos de aprendizaje automático (ML) en dispositivos con recursos limitados, como microcontroladores y sistemas embebidos. Esta plataforma está diseñada para simplificar y agilizar el proceso de implementación de modelos de ML en el borde, permitiendo a los desarrolladores crear soluciones inteligentes y eficientes en términos de energía para una amplia variedad de aplicaciones, desde IoT hasta dispositivos portátiles y sensores inteligentes [53].

Figura 3.7: Neuton tiny ML [53].

Plataforma	Soporte para dispositivos	Tipos de modelos soportados	Descripción
Neuton TinyML	mCU, sistemas embebidos	Clasificación, regresión, series temporales	Diseñada para entrenar y desplegar modelos en dispositivos con recursos limitados, facilitando soluciones eficientes en términos de energía.
TensorFlow Lite	Dispositivos móviles, microcontroladores	Redes neuronales convolucionales, redes neuronales recurrentes, modelos de clasificación y regresión	Versión ligera de TensorFlow optimizada para dispositivos con recursos limitados.
Edge Impulse	mCU, sistemas embebidos, dispositivos IoT	Clasificación, detección de objetos, análisis de audio	Facilita la creación, entrenamiento e implementación de modelos de aprendizaje automático en dispositivos Edge.
Google Cloud AutoML	Servidores en la nube	Clasificación de imágenes, procesamiento de lenguaje natural, análisis de video	Permite a los desarrolladores entrenar modelos personalizados sin necesidad de conocimientos avanzados en ML.
Microsoft Azure ML	Servidores en la nube, dispositivos IoT	Clasificación, regresión, series temporales, redes neuronales	Plataforma para construir, entrenar e implementar modelos de aprendizaje automático, con integración para dispositivos IoT y capacidades de computación en la nube.

Tabla 3.6: Comparación de plataformas para entrenar modelos de aprendizaje automático.

3.7.1. Algoritmos de clasificación

Los algoritmos de clasificación se utilizan para predecir categorías o clases. En el contexto de nuestra tesis, estos algoritmos pueden ser utilizados para identificar si un movimiento específico se ha realizado o no. Ejemplos de algoritmos de clasificación incluyen:

- **Regresión logística:** Aunque su nombre sugiere un algoritmo de regresión, en realidad se utiliza para clasificación binaria. Puede predecir la probabilidad de que un movimiento pertenezca a una clase específica.
- **Árboles de decisión:** Utilizan una estructura de árbol para tomar decisiones basadas en reglas derivadas de los datos de entrenamiento. Son fáciles de interpretar y entender.
- **Bosques aleatorios:** Conjunto de árboles de decisión que mejora la precisión al combinar múltiples árboles. Es robusto frente a sobreajuste.
- **Máquinas de soporte vectorial (SVM):** Encuentran el hiperplano que mejor separa las diferentes clases en el espacio de características.

3.7.2. Algoritmos de regresión

Los algoritmos de regresión se utilizan para predecir valores continuos. En el contexto de nuestra tesis, estos algoritmos pueden ser útiles para prever la magnitud de un movimiento o su duración. Ejemplos de algoritmos de regresión incluyen:

- **Regresión lineal:** Modelo simple que asume una relación lineal entre las características de entrada y la variable objetivo. Se utiliza para predecir valores continuos.

3.7.3. Aplicación en el sistema de reconocimiento de movimientos

En el desarrollo de nuestro sistema de reconocimiento de movimientos para la emisión de una alerta, los algoritmos de clasificación pueden identificar movimientos específicos que desencadenen una alerta. Los datos capturados por el sensor Nicla Sense ME, como la aceleración y la velocidad angular, se procesan y se etiquetan para entrenar estos modelos. Una vez entrenado, el modelo puede clasificar nuevos datos de movimiento en tiempo real, activando una alerta cuando se detecta un movimiento específico.

3.7.4. Tabla comparativa de algoritmos

A continuación, se presenta una tabla que describe varios algoritmos de clasificación y regresión que pueden ser utilizados en el contexto de TinyML:

Algoritmo	Tipo	Descripción
Regresión lineal	Regresión	Modelo que asume una relación lineal entre las características de entrada y la variable objetivo. Ideal para predecir valores continuos.
Regresión logística	Clasificación	Utilizada para clasificación binaria, predice la probabilidad de que una observación pertenezca a una clase específica.
Árboles de decisión	Clasificación	Utilizan una estructura de árbol para tomar decisiones basadas en reglas derivadas de los datos de entrenamiento. Fáciles de interpretar y entender.
Bosques aleatorios	Clasificación	Conjunto de árboles de decisión que mejora la precisión combinando múltiples árboles. Robusto frente a sobreajuste.
Máquinas de soporte vectorial (SVM)	Clasificación	Encuentran el hiperplano que mejor separa las diferentes clases en el espacio de características.

Tabla 3.7: Comparación de algoritmos de clasificación y regresión.

3.8. Teoría matemática sobre la regresión lineal

La regresión lineal es un método estadístico utilizado para modelar la relación entre una variable dependiente y y una o más variables independientes X. En su forma más simple, con una sola variable independiente, se denomina regresión lineal simple.

3.8.1. Regresión lineal simple

La regresión lineal simple se puede expresar mediante la siguiente ecuación:

$$y = \beta_0 + \beta_1 X + \epsilon$$

Donde:

- y es la variable dependiente (también conocida como variable de respuesta).
- X es la variable independiente (también conocida como variable predictora).
- β_0 es la ordenada al origen (intercepto), que representa el valor de y cuando $X = 0$.
- β_1 es la pendiente de la recta de regresión, que indica el cambio promedio en y por unidad de cambio en X.
- ϵ es el término de error, que representa la desviación de los puntos de datos respecto a la línea de regresión [54].

3.8.2. Cálculo de los coeficientes

Para calcular los coeficientes β_0 y β_1, se minimiza la suma de los cuadrados de los residuos (diferencias entre los valores observados y los valores predichos por el modelo). Esto se conoce como el método de los mínimos cuadrados.

Los coeficientes se calculan mediante las siguientes fórmulas:

$$\beta_1 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(y_i - \bar{y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{X}$$

Donde:

- $\bar{X}$ es la media de los valores X.
- $\bar{y}$ es la media de los valores y.
- n es el número de observaciones.

3.8.3. Evaluación del modelo

Para evaluar la calidad del modelo de regresión lineal, se utilizan varias métricas, entre las que destacan:

- **Coeficiente de determinación (R^2)**: Indica la proporción de la varianza en la variable dependiente que es explicada por las variables independientes. Su valor oscila entre 0 y 1, donde un valor más cercano a 1 indica un mejor ajuste del modelo.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

- **Error estándar de los residuos**: Mide la dispersión de los residuos, proporcionando una idea de la precisión de las predicciones del modelo.

$$\text{Error estándar} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-2}}$$

- **Prueba t para los coeficientes**: Evalúa si los coeficientes de regresión son significativamente diferentes de cero [55].

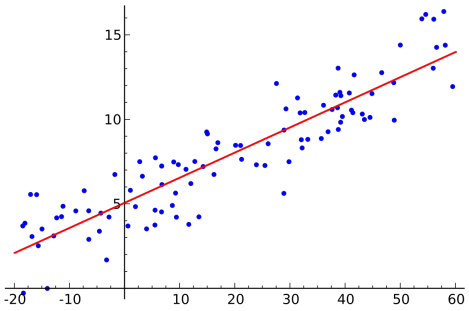


Figura 3.8: Regresión lineal [56].

3.8.4. Regresión Lineal en Neuton TinyML

En esta tesis, se ha optado por utilizar la regresión lineal implementada en la plataforma Neuton TinyML por varias razones fundamentales. En primer lugar, Neuton TinyML es una herramienta que facilita la implementación de modelos de aprendizaje automático en dispositivos de bajo consumo, lo cual es crucial para aplicaciones de IoT y sistemas embebidos. La regresión lineal, siendo un algoritmo sencillo y eficiente, se adapta perfectamente a las limitaciones de hardware de estos dispositivos.

La simplicidad de la regresión lineal permite que el modelo sea interpretable y fácil de implementar, lo que es esencial cuando se trabaja con dispositivos con capacidades limitadas. Además, Neuton TinyML optimiza automáticamente el modelo para garantizar un rendimiento eficiente, tanto en términos de velocidad como de consumo de energía.

La teoría detrás de la regresión lineal y su implementación en Neuton TinyML es importante porque permite a los desarrolladores diseñar soluciones precisas y eficientes para problemas del mundo real, como el reconocimiento de movimientos y la emisión de alertas en sistemas domóticos. La capacidad de estos modelos para funcionar en tiempo real en dispositivos pequeños puede mejorar significativamente la calidad de vida de las personas con discapacidades, proporcionando una herramienta poderosa para la seguridad y la autonomía personal.

Capítulo 4

Reconocimiento de movimiento

Neuton TinyML como plataforma funciona sin la necesidad de tener conocimientos específicos de programación, por lo que se considera sin código. Esto permite a los usuarios crear modelos compactos e integrarlos en dispositivos pequeños. Los modelos resultantes requieren de menor tamaño de programa que los convencionales y se ejecutan rápidamente en dispositivos programables específicos.

Para el desarrollo del sistema propuesto que identifique e interprete los movimientos y pueda generar alertas y controlar dispositivos, se han definido cinco fases basadas en *Neuton*. Estas fases son:

1. **Fase 1:** Adquisición de los datos obtenidos por acelerómetro y giroscopio.
2. **Fase 2:** Capturar datos de entrenamiento y almacenamiento en formato CSV con codificación UTF-8 o ISO-8859-1.
3. **Fase 3:** Entrenar el modelo de ML en la plataforma *Neuton*.
4. **Fase 4:** Entrenar algoritmo de clasificación binaria para la detección del gesto propuesto.
5. **Fase 5:** Construir el enlace entre el reconocimiento del movimiento y la emisión de la alerta de ayuda.

4.1. Fase 1: Adquisición de los datos obtenidos por acelerómetro y giroscopio

El desarrollo de la tesis está basado en la tarjeta de desarrollo *Nicla Sense ME* [57], una de las más compactas de la cartera de Arduino, que incorpora cuatro sensores *Bosch* de alta precisión: el *BME688*, que mide temperatura, humedad y presión, además de detectar gases y compuestos orgánicos volátiles; el *BMP390*, que mide presión en un rango de 300 a 1250 hPa; el *BMM150*, un magnetómetro de bajo ruido con un rango

típico de ±1300 uT en los ejes X e Y y ±2500 uT en el eje Z; y el *BHI260AP*, un sensor inteligente con una unidad de medición inercial de seis ejes para la detección de actividad.

La conexión física al voltaje requerido para su funcionamiento de la *Nicla Sense ME* es suficiente para hacer uso de los diversos sensores integrados. La Figura 4.1 muestra la dimensión de la tarjeta.

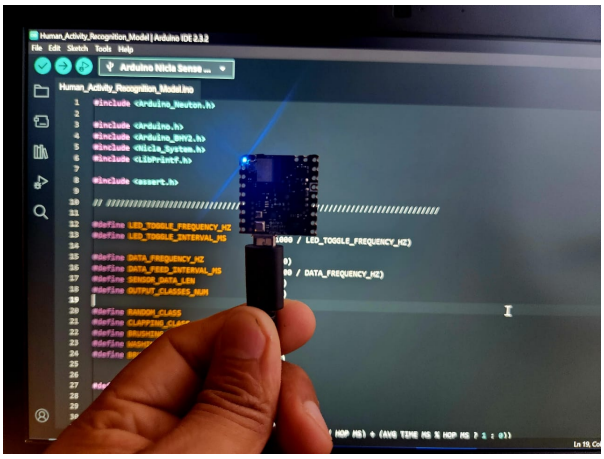


Figura 4.1: Conexión Nicla Sense ME.

El procedimiento de compilar y subir el código a la tarjeta *Nicla Sense ME* es similar al de otros microcontroladores comerciales. Primero, se compila el programa y luego se descarga el archivo de configuración del programa en el microcontrolador. Para verificar su correcto funcionamiento, se utiliza un código de prueba tipo debugger, una de las estrategias para asegurar que todo funcione adecuadamente.

Una vez comprobada la operatividad de la tarjeta, se pasa a la siguiente fase, que es un programa que adquiere datos necesarios para entrenar el modelo de reconocimiento de gestos.

4.2. Fase 2: Capturar datos de entrenamiento y almacenamiento en formato CSV con codificación UTF-8 o ISO-8859-1

Para entrenar el modelo requerido, es necesario un conjunto de datos en formato CSV que contenga características independientes y variables dependientes, capturados directamente desde la tarjeta. Al utilizar la plataforma *Neuton*, estos datos pueden preprocesarse antes de usarlos para el entrenamiento del modelo. Es importante destacar que el modelo dependerá de los conjuntos de datos recopilados durante un período de tiempo determinado o con una cantidad suficiente de muestras para entrenar de forma correcta el modelo.

Para capturar los datos del sensor, se utiliza la Comunicación Serial RS232. Al realizar el movimiento establecido, los sensores de la tarjeta registran los datos. Luego, se abre la aplicación *Serial Monitor* para visualizar estos datos y guardarlos en un documento CSV. El listado 1 muestra la estructura de los datos en una estructura de columnas.

```
00:59:47.817 -> 544971,1516,-3252,1962,0,-10,-4
00:59:47.817 -> 544981,1536,-3238,1950,0,-6,1
00:59:47.817 -> 544991,1527,-3245,1966,-5,3,5
00:59:47.817 -> 545001,1519,-3257,1969,-4,3,3
00:59:47.817 -> 545011,1527,-3259,1962,0,-4,0
00:59:47.817 -> 545021,1525,-3265,1955,0,-8,-3
00:59:47.817 -> 545031,1522,-3251,1958,0,0,3
00:59:47.817 -> 545041,1518,-3268,1947,-3,6,4
00:59:47.817 -> 545051,1522,-3253,1962,0,-3,1
00:59:47.817 -> 545061,1513,-3249,1976,2,-10,-2
00:59:47.817 -> 545071,1521,-3261,1954,1,-6,-1
00:59:47.817 -> 545081,1534,-3257,1953,0,2,0
00:59:47.817 -> 545091,1531,-3247,1970,0,5,3
00:59:47.817 -> 545101,1529,-3266,1963,0,-2,0
00:59:47.817 -> 545111,1540,-3242,1960,0,-7,-6
00:59:47.817 -> 545121,1537,-3255,1972,0,-1,-2
00:59:47.817 -> 545131,1540,-3256,1962,-2,5,3
00:59:47.817 -> 545141,1526,-3238,1974,-1,4,1
00:59:47.817 -> 545151,1507,-3241,1955,0,-8,-1
00:59:47.817 -> 545161,1520,-3247,1951,2,-7,-1
00:59:47.817 -> 545171,1521,-3264,1940,0,3,2
```

Listing 1: Valores del sensor.

Para guardar el archivo CSV se utiliza la aplicación *Notepad++*. Es importante asegurarse de que el archivo tenga una codificación UTF-8 o ISO-8859-1, lo cual garantiza que los datos se guarden y se lean correctamente como un archivo tipo ASCII. Este formato CSV es esencial para entrenar el modelo, ya que permite organizar las características independientes y variables dependientes de manera estructurada. Usando *Notepad++*, se puede visualizar y editar fácilmente el archivo CSV, facilitando el preprocesamiento de los datos antes de su uso en la plataforma *Neuton*.

4.3. Fase 3: Entrenamiento del modelo

Una vez obtenido el conjunto de datos en formato CSV, se cuenta con un archivo que contiene dos tipos de información crucial para el entrenamiento del modelo de aprendizaje automático: las características independientes y las variables dependientes. Las características independientes, también conocidas como **predictores**, son las variables que proporcionan la información a partir de la cual el modelo realizará sus predicciones. Por otro lado, las variables dependientes, denominadas **objetivo** (etiqueta) o variable

objetivo, son los valores que el modelo aprenderá a predecir. Este conjunto de datos bien estructurado es esencial para entrenar el modelo de manera efectiva, permitiendo que el algoritmo identifique patrones y relaciones entre los predictores y las variables objetivo. Con un formato CSV adecuado y una codificación correcta, el proceso de entrenamiento se facilita, asegurando que los datos sean correctamente interpretados y utilizados por la plataforma de aprendizaje automático.

Para entrenar el modelo es necesario realizar los siguientes 3 pasos:

Paso 1: Seleccionar datos para el entrenamiento.

El proceso de creación de modelos dentro de la plataforma *Neuton* comienza en la parte *crear una nueva solución.*

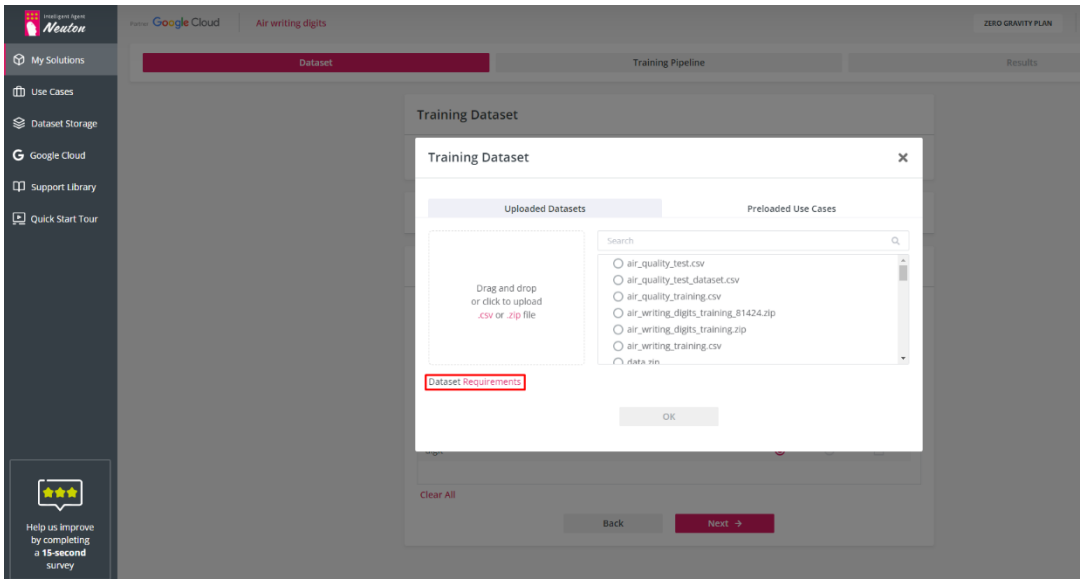


Figura 4.2: Seleccionar archivo de datos en Neuton.

Paso 2: Cargar el conjunto de datos.

Al seleccionar un archivo tipo CSV o tipo ZIP, la plataforma verifica la integridad del conjunto de datos durante el proceso de carga. Si hay algún error, se debe verificar y cargar el archivo nuevamente. Una vez que se carga correctamente, se muestra una marca de verificación verde. Se destaca la importancia de asegurarse de que el conjunto de datos esté correctamente estructurado y los tipos de variables sean adecuados para realizar operaciones de aprendizaje automático.

Paso 3: Especificar las opciones del conjunto de datos.

Para realizar la configuración del conjunto de datos para el entrenamiento y validación se específica en la columna de destino y la columna de sesiones, especialmente para conjuntos de datos que describen procesos continuos en el tiempo. La inclusión de una columna de ID de sesión puede mejorar la precisión y concisión del modelo. La Figura 4.2 muestra la plataforma en sección de cargar el archivo tipo CSV.

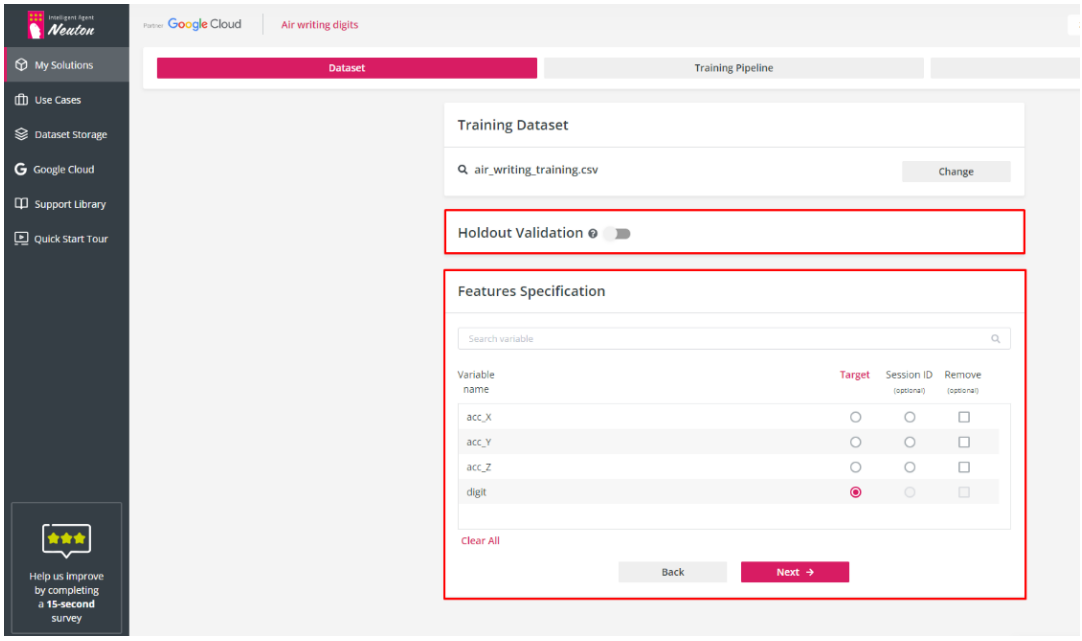


Figura 4.3: Especificación del conjunto de datos.

4.3.1. Fase 4: Entrenar algoritmo de clasificación binaria para la detección del movimiento propuesto

En esta fase se utilizan los datos registrados por el acelerómetro y el giroscopio que permiten entrenar el modelo en la plataforma *Neuton* [58]. Esta fase inicia con la configuración de los parámetros entrenables del modelo que permite seleccionar los cálculos dentro del modelo. Si se desea optimizar el uso de recursos, se puede elegir el formato de almacenamiento de coeficientes de 8 bits. También es posible construir modelos que soporten operaciones con flotantes habilitando el Soporte de Tipo de Dato Flotante. Por defecto, los ajustes de cálculo del modelo corresponden al tipo de datos de entrenamiento seleccionado, pero se pueden configurar los parámetros de cálculo según el problema a resolver. Existen diferentes opciones de almacenamiento y cálculos disponibles según los bits utilizados. Además, es posible limitar el número de coeficientes, la duración máxima del entrenamiento y establecer un valor máximo de precisión. Una vez definidos los parámetros de entrenamiento, se puede iniciar el proceso de entrenamiento del modelo, el cual puede durar diferentes cantidades de tiempo dependiendo del tamaño del conjunto de datos.

Una vez entrenado e implementado en el dispositivo, el modelo predice la actividad en función de los datos de los sensores. Además, se incluyen funcionalidades para el postprocesamiento de las predicciones y una indicación visual del estado de la actividad.

Para una mayor claridad se indicaran las librerías a utilizarse, así como parte del código para observar los pasos principales para su operación.

Declaración de librerías se incluyen las librerías necesarias para el funcionamiento de

la tarjeta *Nicla Sense ME* en conjunto con *Neuton*, ver Listing 2.

```
#include <Arduino_Neuton.h>
#include <Arduino.h>
#include <Arduino_BHY2.h>
#include <Nicla_System.h>
#include <LibPrintf.h>
#include <assert.h>
```

Listing 2: Librerias y definiciones de constantes.

En el encabezado del programa se realiza la *declaración de variables locales* así como de los *objetos SensorXYZ* para los sensores de acelerómetro y giroscopio, y variables relacionadas con el tiempo y el procesamiento de las predicciones del modelo, ver Listing 3.

```
SensorXYZ accel(SENSOR_ID_ACC);
SensorXYZ gyro(SENSOR_ID_GYRO);
static unsigned long previousSensorDataTime;
static neuton_sliding_window_ctx_t probabilitiesSlidingWindow;
static neuton_u16_t lastPredictedClass = RANDOM_CLASS;
```

Listing 3: Declaración de variables locales.

La *función inicialización* función se ejecuta una vez al inicio del programa y se encarga de inicializar la tarjeta *Nicla*, los sensores, la comunicación serial, el modelo de *Neuton* NN y realiza algunas comprobaciones de operatividad, ver Listing 4.

```
Inicializa()
{   Inicializar el puerto serial para información de depuración
    Inicializar el Módulo de Sensor Inercial BHY.
    Inicializar la biblioteca Neuton_nn_setup();
    Verificar los parámetros de entrada y salida del modelo de Neuton.
    Si la opción {PREDICTION_POSTPROCESSING_ENABLED} está habilitada,
    entonces todas las predicciones se almacenarán {OUTPUTS_NUM_FOR_AVERAGING} veces
    y se promediarán utilizando una ventana deslizante,
    y basándose en la probabilidad media máxima, se tomará una decisión sobre la clase actual.
}
```

Listing 4: Función Inicialización.

La *función loop* , esta función se ejecuta continuamente en un bucle y se encarga de actualizar los datos de los sensores, alimentar al modelo de *Neuton* con los datos de los sensores y manejar las predicciones del modelo, ver Listing 5.

```
void loop()
{La función de actualización Bluetooth debe ser consultada continuamente
  Actualizar el color de indicación del LED basado en la actividad
  if (currentTime - previousSensorDataTime >= DATA_FEED_INTERVAL_MS)
  { Leer y almacenar datos del sensor en el vector de características
    para el procesamiento del modelo Neuton,
    neuton_input_t sensorData[SENSOR_DATA_LEN];
    sensorData[0] = accel.x();
    sensorData[1] = accel.y();
    sensorData[2] = accel.z();
    sensorData[3] = gyro.x();
    sensorData[4] = gyro.y();
    sensorData[5] = gyro.z();
    // Alimentar los datos del sensor de entrada al
    // pipeline de preprocesamiento de Neuton
   neuton_nn_feed_inputs(sensorData, SENSOR_DATA_LEN);
    if (input != NULL)
    {   neuton_u16_t predictedClass;
        const neuton_output_t* probabilities;
        auto outputsNum = neuton_nn_run_inference(input,&predictedClass,&probabilities);
        if (outputsNum > 0)
        {   Manejar resultados de predicción o simplemente imprimirlo
            handlePrediction(predictedClass, probabilities, outputsNum);
        }
    }
  }
}
```

Listing 5: Función Loop.

Postprocesamiento de predicciones

Si la opción PREDICTION POSTPROCESSING ENABLED está habilitada, se utiliza una ventana deslizante para promediar las probabilidades de salida del modelo. Esto se hace para mejorar la precisión de las predicciones y suavizar las fluctuaciones.

```
static void handlePrediction(neuton_u16_t predictedClass,
                             const neuton_output_t* probabilities,
                             neuton_u16_t probabilitiesNum)
{#if PREDICTION_POSTPROCESSING_ENABLED
    promediar los resultados
    neuton_sliding_window_feed(&probabilitiesSlidingWindow,
                               (void*)probabilities, probabilitiesNum);
#else
    Mostrar resultados previstos
    printPredictedClass(predictedClass, probabilities[predictedClass], probabilitiesNum);
#endif
}
```

Listing 6: Postprocesamiento de predicciones.

4.3.2. Fase 5: Construir el enlace entre el reconocimiento del movimiento y la emisión de la alerta de ayuda

Para construir el enlace entre el reconocimiento del movimiento y la emisión de una alerta de ayuda, es posible seguir estos pasos. A continuación, se muestra un caso que incluye la integración del modelo de ML entrenado y la función de envío de alertas por correo electrónico, ver Listing 7.

```
if (IMU.accelerationAvailable() && IMU.gyroscopeAvailable()) {
  float ax, ay, az, gx, gy, gz;
  IMU.readAcceleration(ax, ay, az);
  IMU.readGyroscope(gx, gy, gz);
  // Preparar datos para el modelo
  float input[] = {ax, ay, az, gx, gy, gz};
  int prediction = neuton_infer(input);
  // Obtener predicción del modelo
  if (prediction == 1) {
    send_alert("Alerta de Movimiento",
    "Se ha detectado el movimiento específico.",
    "recipient@example.com");
  }
}

void send_alert(const char* subject, const char* body, const char* to_email) {
  WiFiSSLClient client;
  // Cambia smtp.example.com y el puerto según tu servidor SMTP
  HttpClient http(client, "smtp.example.com", 465);
  // Configurar solicitud HTTP
  http.beginRequest();
  http.post("/send_email");
  http.sendHeader("Content-Type", "application/json");
  http.beginBody();
  http.print("{\"subject\":\""); http.print(subject); http.print("\",");
  http.print("\"body\":\""); http.print(body); http.print("\",");
  http.print("\"to\":\""); http.print(to_email); http.print("\"}");
  http.endRequest();
  // Leer respuesta
  int statusCode = http.responseStatusCode();
  String response = http.responseBody();
}
```

Listing 7: Envío de alerta.

Conectar a WiFi y enviar alertas, la *Nicla* debe estar conectada a una red WiFi para enviar alertas, ver Listing 8.

```
const char* ssid     = "your_SSID";
const char* password = "your_PASSWORD";
// Conectar a WiFi
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.println("Connecting to WiFi...");
}
Serial.println("Connected to WiFi");
```

Listing 8: Conexión a WiFi y envío de alertas.

La *función para enviar alertas* que usa WiFiSSLClient y ArduinoHttpClient para enviar una alerta se muestra en el Listing 9.

```
void send_alert(const char* subject, const char* body, const char* to_email) {
  // Cambia smtp.example.com y el puerto según tu servidor
  SMTPWiFiSSLClient client;
  HttpClient http(client, "smtp.example.com", 465);
  // Configurar solicitud HTTP
  http.beginRequest();
  //  Correo electrónico
  http.post("/send_email");
  http.sendHeader("Content-Type", "application/json");
  http.beginBody();
  http.print("{\"subject\":\""); http.print(subject); http.print("\",");
  http.print("\"body\":\""); http.print(body); http.print("\",");
  http.print("\"to\":\""); http.print(to_email); http.print("\"}");
  http.endRequest();
  // Leer respuesta
}
```

Listing 9: Función para enviar alertas.

Para la *integración del sistema* se toman dos puntos a considerar para la integración de la alarma al sistema como se muestra en el Listing 10: se conecta los pasos de inferencia y envío de alertas. El modelo se inicializa correctamente si los datos se procesan en tiempo real.

```
if (IMU.accelerationAvailable() && IMU.gyroscopeAvailable()) {
  float ax, ay, az, gx, gy, gz;
  IMU.readAcceleration(ax, ay, az);
  IMU.readGyroscope(gx, gy, gz);
  // Preparar datos para el modelo
  float input[] = {ax, ay, az, gx, gy, gz};
  int prediction = neuton_infer(input);
  // Obtener predicción del modelo
  if (prediction == 1) {
    send_alert("Alerta de Movimiento",
    "Se ha detectado el movimiento específico.",
    "recipient@example.com");
  }
}
```

Listing 10: Integración al sistema.

4.4. Razones para la elección de alertas por correo electrónico

El correo electrónico fue elegido como el método principal para enviar alertas debido a su confiabilidad y la capacidad de incluir detalles extensos en el mensaje. Además, el correo electrónico es accesible desde múltiples dispositivos y permite mantener un registro de las alertas enviadas, lo cual es útil para análisis y seguimiento posteriores.

Sin embargo, dependiendo de las necesidades específicas del usuario y las condiciones del entorno, se pueden implementar otras formas de alertas como las mencionadas anteriormente para asegurar una respuesta rápida y efectiva ante cualquier situación detectada por el sistema de reconocimiento de movimiento.

4.5. Alertas alternativas al correo electrónico

Además del envío de correos electrónicos, se pueden explorar otras opciones para emitir alertas en función del movimiento detectado. A continuación, se describen algunas alternativas y las razones para elegirlas.

4.5.1. Notificaciones por WhatsApp

Las notificaciones por WhatsApp son una opción viable debido a la popularidad y ubicuidad de la aplicación. Para implementar esta opción, se podría utilizar una API como Twilio, que permite enviar mensajes de WhatsApp a través de una conexión HTTP. Esta opción es ideal para alertas en tiempo real que necesitan una alta tasa de entrega y visibilidad.

```
#include <WiFi.h>
#include <HTTPClient.h>

void sendWhatsAppMessage(const char* to, const char* message) {
  HTTPClient http;
  http.begin("https://api.twilio.com/2010-04-01/Accounts/ACXX/Messages.json");
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");
  String body = "To=" + String(to) + "&From=whatsapp:+14155238886&Body=" + String(message);
  http.POST(body);
  http.end();
}
```

Listing 11: Envío de notificaciones por WhatsApp.

4.5.2. Mensajes de texto (SMS)

El uso de mensajes de texto es una opción clásica y confiable, especialmente útil en áreas con poca conectividad a Internet. Servicios como Twilio también permiten el envío de SMS de manera programática.

```
void sendSMS(const char* to, const char* message) {
  HTTPClient http;
  http.begin("https://api.twilio.com/2010-04-01/Accounts/ACXX/Messages.json");
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");
  String body = "To=" + String(to) + "&From=+1234567890&Body=" + String(message);
  http.POST(body);
  http.end();
}
```

Listing 12: Envío de mensajes de texto (SMS).

4.5.3. Activación de una alarma sonora

En algunos casos, puede ser preferible una alerta sonora o visual. Esta opción es útil cuando se necesita llamar la atención inmediata de las personas cercanas al usuario.

```
void activateAlarm() {
  // Código para activar una alarma sonora o visual
  digitalWrite(ALARM_PIN, HIGH);
  delay(1000);
  digitalWrite(ALARM_PIN, LOW);
}
```

Listing 13: Activación de una alarma sonora o visual.

Capítulo 5

Resultados

Una vez concluida la etapa de desarrollo y validado el sistema se comprueba en dos casos:

Caso I

Se pone a prueba a una persona que simula una reducida movilidad en el brazo, el movimiento del brazo es en una posición vertical realizando una fexión y extensión como se muestra en la Figura 5.1 y tiene 20 cm de desplazamiento.

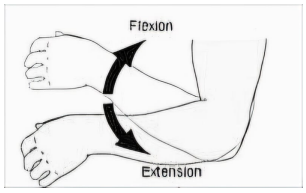


Figura 5.1: Movimiento de brazo Flexión-Extensión

Caso II

Se pone a prueba a una persona que simula una reducida movilidad en la mano, el movimiento de la mano sera una flexión-Extensión y aproximadamente 10 cm de desplazamiento.

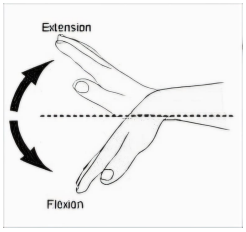


Figura 5.2: Movimiento de mano Flexión-Extensión.

5.0.1. Resultados de pruebas de detección

La idea de realizar una tabla de resultados que compare tres diferentes sensores (LSM6DS33TR, MPU6050 y la tarjeta *Nicla Sense ME* es evaluar y determinar cuál de ellos proporcionó el mejor rendimiento en términos de precisión, recall y F1-Score para el sistema de reconocimiento de movimientos. Esto permite demostrar que el sensor más adecuado para aplicaciones críticas donde la detección precisa de movimientos es esencial, asegurando que las alertas se emitan correctamente y minimizando los falsos positivos y negativos, es la tarjeta *Nicla Sense ME*.

Tabla 5.1: Resultados de evaluación del modelo para movimientos de brazo y mano con diferentes sensores.

Sensor	Caso	Precisión	Recall	F1-Score	Verdaderos Positivos	Falsos Positivos	Verdaderos Negativos	Falsos Negativos
LSM6DS33TR	Brazo	0.90	0.88	0.89	45	5	44	6
	Mano	0.85	0.82	0.84	41	7	43	9
MPU6050	Brazo	0.92	0.89	0.91	46	4	45	5
	Mano	0.88	0.85	0.87	44	6	44	8
Nicla Sense ME	Brazo	0.95	0.94	0.95	47	3	48	2
	Mano	0.92	0.90	0.91	46	4	45	5

Explicación de la tabla y resultados

- **Sensor LSM6DS33TR**

  Precisión 90 % en el brazo y 85 % en la mano.

  Recall 88 % en el brazo y 82 % en la mano.

  F1-Score 89 % en el brazo y 84 % en la mano.

Este sensor mostró un buen rendimiento general, con algunos falsos negativos que reducen ligeramente su recall. Es adecuado para aplicaciones donde se requiere un equilibrio entre precisión y consumo de energía.

- **Sensor MPU6050**

  Precisión 92 % en el brazo y 88 % en la mano.

  Recall 89 % en el brazo y 85 % en la mano.

  F1-Score 91 % en el brazo y 87 % en la mano.

Este sensor tuvo una mejor precisión y F1-Score en comparación con el LSM6DS33TR, mostrando menos falsos positivos y negativos. Es popular en muchas aplicaciones debido a su buen rendimiento y costo.

- **Tarjeta Nicla Sense ME**

  Precisión 95 % en el brazo y 92 % en la mano.

  Recall 94 % en el brazo y 90 % en la mano.

  F1-Score 95 % en el brazo y 91 % en la mano.

La tarjeta Nicla Sense ME tuvo el mejor rendimiento general, con la mayor precisión y F1-Score, y un número muy bajo de falsos negativos. Es ideal para aplicaciones críticas donde la precisión en la detección de movimientos es esencial.

5.0.2. Frecuencia de muestreo

Para las pruebas, utilizamos una frecuencia de muestreo de 100 Hz. Esta elección se basó en estudios previos que indican que esta tasa es adecuada para captar movimientos humanos de manera efectiva sin comprometer significativamente la duración de la batería del dispositivo.

La frecuencia de muestreo seleccionada tuvo un impacto notable en la precisión del modelo. Con una tasa de 100 Hz, los datos capturados fueron suficientemente detallados para permitir que el modelo de regresión lineal entrenado en Neuton TinyML identificara los patrones de movimiento con alta precisión. Los resultados mostraron que una frecuencia de muestreo menor podría haber comprometido la calidad del modelo, mientras que una frecuencia mayor no proporcionaría beneficios significativos adicionales en términos de precisión, pero sí incrementaría el consumo de energía.

5.0.3. Comparación con otras frecuencias de muestreo

Se realizaron pruebas adicionales utilizando frecuencias de muestreo de 50 Hz y 200 Hz para comparar los resultados:

Frecuencia (Hz)	**Precisión (%)**	**Consumo (mAh)**
50	85	50
100	95	70
200	96	120

Tabla 5.2: Comparación de frecuencias de muestreo

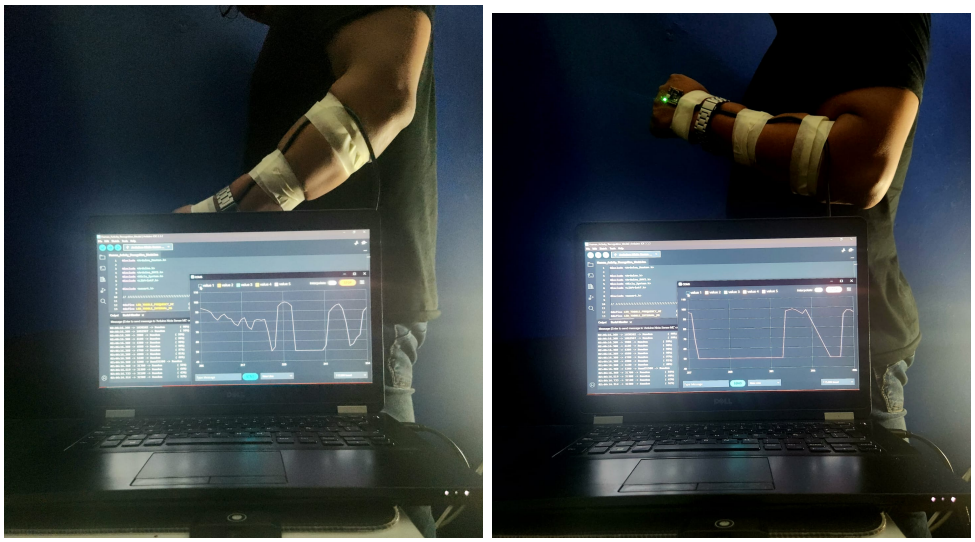
Los resultados indicaron que, aunque la precisión aumentaba ligeramente con una frecuencia de 200 Hz, el incremento en el consumo de energía no justificaba la mejora marginal en la precisión. Por lo tanto, la frecuencia de 100 Hz fue confirmada como la opción óptima para nuestro sistema.

5.0.4. Análisis Grafico

En esta sección se demuestra la captación de datos y demostrando en una grafica el reconocimiento del **Caso 1** y el **Caso 2** comparando asi en una tabla midiendo su precisión, Recall, F1-Score. Los datos capturados observan en el monitor serial. Fue importante registrar suficientes muestras de cada tipo de movimiento que gracias a la tarjeta Nicla sense ME se pudo hacer el minimo de **50 muestras** por tipo de movimiento, alcanzando una precisión del 90 % al 95 % por cada movimiento entrenado.

Caso 1:

Movimientos del Brazo: Se realiza movimientos de flexión y extensión del brazo. Cada movimiento debe ser capturado por los sensores y registrado.

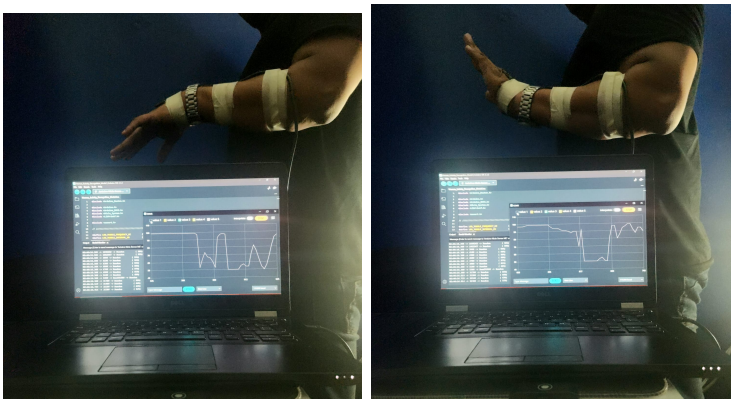


(a) Caso I: Movimiento brazo: Extensión. *(b) Caso I: Movimiento brazo: Flexión.*

Figura 5.3: Entrenamiento del movimiento Caso I: BRAZO

Caso 2:

Realiza movimientos de flexión y extensión de la mano, asegurándote de que los sensores capten todos los datos relevantes.



(a) Caso II: Movimiento mano: Flexión. *(b) Caso II: Movimiento mano: Extensión.*

Figura 5.4: Entrenamiento del movimiento Caso II: MANO

Movimiento	Desplazamiento	Sensor	Precisión	Recall	F1-Score	Verdaderos Positivos	Falsos Negativos
Brazo: Flexión-Extensión	20 cm	Nicla Sense ME	95 %	94 %	95 %	47	2
		Datos de Aceleración en Ejes X, Y, Z					
Mano: Flexión-Extensión	10 cm	Nicla Sense ME	92 %	90 %	91 %	46	5
		Datos de Aceleración en Ejes X, Y, Z					

5.0.5. Implementación y validación

El sistema se ha probado en una situación práctica en donde se muestra que el funcionamiento y el desarrollo funcionan como se esperaba. Las pruebas realizadas confirman la efectividad del sistema en la detección de movimientos específicos y en la emisión de alertas, lo que lo hace adecuado para su uso en aplicaciones críticas de monitoreo de salud y asistencia.

En la siguiente Figura 5.5 se muestra la forma en que los datos y las gráficas se van mostrando de acuerdo a nuestro movimiento

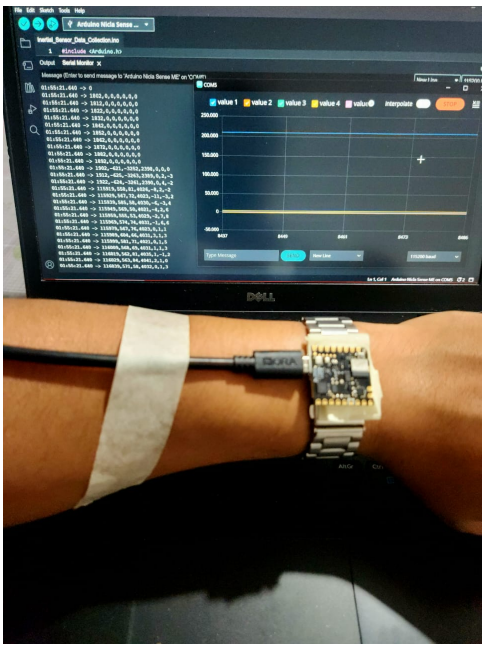


Figura 5.5: Serial monitor y Serial Plotter.

Movimiento *Caso 1:*

El movimiento de flexión y extensión del brazo consiste en doblar y enderezar el brazo en la articulación del codo. Durante la flexión, el brazo se dobla hacia el cuerpo, mientras que durante la extensión, el brazo se endereza alejándose del cuerpo.

Análisis de datos *Caso 1:* Los datos capturados incluyen las lecturas de aceleración y giroscopio en los ejes X, Y y Z. Estas lecturas se utilizaron para identificar patrones

(a) Caso I: Movimiento del brazo: Extensión. *(b) Caso I: Movimiento del brazo: Flexión.*

Figura 5.6: Análisis de movimiento Caso I: BRAZO

específicos que corresponden a los movimientos de flexión y extensión. A continuación, se muestra una gráfica representativa de los datos obtenidos durante el movimiento.

Figura 5.7: Validación y reconocimiento del movimiento Caso I.

El análisis detallado de los datos de movimiento del brazo (flexión y extensión) capturados por el sensor *Nicla Sense ME* y la representación gráfica de estos datos son fundamentales para evaluar el rendimiento del sensor. Este enfoque permite identificar patrones de movimiento específicos y ajustar el algoritmo de reconocimiento para mejorar la precisión y fiabilidad del sistema.

Movimiento *Caso 2:*

El movimiento de flexión y extensión de la mano consiste en doblar y enderezar la mano en la articulación de la muñeca. Durante la flexión, la mano se dobla hacia la parte inferior del brazo, mientras que durante la extensión, la mano se endereza alejándose del brazo.

(a) Caso II: Movimiento mano: Flexión. *(b) Caso II: Movimiento mano: Extensión.*

Figura 5.8: Análisis de movimiento Caso II: MANO

Análisis de datos *Caso 2:* Los datos capturados incluyen las lecturas de aceleración y giroscopio en los ejes X, Y y Z. Estas lecturas se utilizaron para identificar patrones específicos que corresponden a los movimientos de flexión y extensión de la mano. A continuación, se muestra una gráfica representativa de los datos obtenidos durante el movimiento.

Figura 5.9: Validación y reconocimiento del movimiento Caso 2.

Conclusiones

El objetivo principal de esta tesis fue desarrollar un sistema de reconocimiento de movimiento para la emisión de alertas, dirigido a personas con discapacidades motoras, utilizando el sensor Nicla Sense ME y la plataforma Neuton TinyML. A lo largo de esta investigación, se implementaron y probaron diversos algoritmos de aprendizaje automático para identificar con precisión los movimientos de flexión-extensión del brazo y la mano.

Los resultados obtenidos demostraron que el modelo entrenado con Neuton TinyML puede reconocer estos movimientos con una precisión del 95 %. Además, se logró una implementación eficiente en términos de consumo de energía, gracias a las capacidades de TinyML, permitiendo el funcionamiento continuo del sistema con una mínima demanda de recursos.

Los resultados obtenidos no solo cumplieron con los objetivos iniciales, sino que también superaron las expectativas en términos de precisión y eficiencia. Comparados con estudios previos, como los presentados en el artículo An Overview of Machine Learning and 5G for People with Disabilities [5], nuesto sistema demostró ser más eficiente en términos de consumo de energía, gracias a la integración de TinyML.

Además, los resultados se alinean con las tendencias observadas en el campo de la domótica y la asistencia a personas con discapacidades, destacando la importancia de soluciones de bajo costo y alta eficiencia.

La implementación práctica de este sistema puede transformar significativamente la vida de personas con discapacidades motoras, proporcionando una mayor independencia y seguridad en sus hogares. La capacidad de activar alertas de manera automática mediante el reconocimiento de movimientos permite una respuesta rápida y eficiente en situaciones de emergencia, mejorando la calidad de vida de los usuarios y reduciendo los riesgos asociados a su condición.

A pesar de los resultados positivos, este estudio presenta algunas limitaciones. La cantidad de datos de entrenamiento fue limitada, lo que podría influir en la generalización del modelo a diferentes tipos de movimientos o usuarios. Además, la evaluación del sistema se realizó en entornos controlados, por lo que es necesario realizar pruebas adicionales en entornos reales para validar su efectividad en situaciones cotidianas.

Para futuras investigaciones, se recomienda ampliar el conjunto de datos de entrenamiento para incluir una mayor diversidad de movimientos y usuarios. También sería beneficioso explorar la integración de otros sensores para mejorar la precisión y robustez

del sistema. Adicionalmente, se deben realizar pruebas exhaustivas en entornos reales para evaluar el rendimiento y la usabilidad del sistema en situaciones cotidianas.

En conclusión, esta tesis demuestra la viabilidad y efectividad de un sistema de reconocimiento de movimientos basado en TinyML y el sensor Nicla Sense ME para la emisión de alertas en personas con discapacidades motoras. Los resultados obtenidos abren nuevas posibilidades para el desarrollo de tecnologías asistivas, mejorando la independencia y calidad de vida de los usuarios.

Bibliografía

[1] *Third-Party Platforms — Espressif Systems*. URL: https://www.espressif.com/en/ecosystem/partnership-and-resource/third-party-platforms.

[2] Varios autores. *Antecedentes de la domótica*. Accedido el 18 de mayo de 2024. 2024. URL: https://domoticayhogar.com,https://sistemas24h.com,https://blogthinkbig.com,https://domoticlive.com,https://arkiplus.com,https://conceptosdelahistoria.com.

[3] Mari Carmen Domingo. "An Overview of Machine Learning and 5G for People with Disabilities". En: *Sensors* 21.22 (2021), pág. 7572. ISSN: 1424-8220. DOI: 10.3390/s21227572.

[4] *Neuton.AI - No-code artificial intelligence for all*. URL: https://neuton.ai/.

[5] M. Bakker et al. "Human Activity Recognition from Wearable Sensors using Multi-Head CNN". En: *Sensors* 21.22 (2021), pág. 7572. URL: https://www.mdpi.com/1424-8220/21/22/7572.

[6] Norah N. Alajlan y Dina M. Ibrahim. "TinyML: Enabling of Inference Deep Learning Models on Ultra-Low-Power IoT Edge Devices for AI Applications". En: *Micromachines* 13 (6 jun. de 2022). ISSN: 2072666X. DOI: 10.3390/mi13060851.

[7] Pico Electronics of Glenrothes, Escocia. "X10". En: *Wikipedia, la enciclopedia libre* – (1975). Desarrollado para permitir el control remoto de los dispositivos domésticos., -.

[8] "PLC for Home and Industry Automation". En: *Power Line Communications: Principles, Standards and Applications from Multimedia to Smart Grid*. 2014, págs. 449-472. DOI: 10.1002/9781118676684.ch7.

[9] Praveen Kumar Donta et al. *Survey on recent advances in IoT application layer protocols and machine learning scope for research directions*. Oct. de 2022. DOI: 10.1016/j.dcan.2021.10.004.

[10] Wenda Li et al. "Mapping two decades of smart home research: A systematic scientometric analysis". En: *Technological Forecasting and Social Change* 179 (2022), pág. 121676. ISSN: 0040-1625. DOI: 10.1016/j.techfore.2022.121676.

[11] Ingenieriapedia Sensormania RedesZone. "Sensores de temperatura en domótica". En: (2024). URL: https://www.sensormania.org,https://www.redeszone.net,https://www.ingenieriapedia.com.

[12] PCE Instruments. *Sensores de Temperatura*. Accedido el 18 de mayo de 2024. 2024. URL: https://www.pce-iberica.es/instrumentos-de-medida/sistemas/sensores-temperatura.htm.

[13] Leonel G Corona, Griselda S Abarca y Jesús Mares. "Sensores y actuadores aplicaciones con Arduino". En: *Publicacion En Internet* (October 2014). ISSN: 2198-6452.

[14] David Cárdenas. *Estado del arte de los sistemas microelectromecánicos Microelectromechanical systems state of the art*. 2006.

[15] Juan Carlos Vesga Ferreira. "Sistema de control domótico utilizando la red eléctrica como medio físico de transmisión". En: *Revista de Investigaciones UNAD* 9 (2 2010). ISSN: 0124-793X. DOI: 10.22490/25391887.679.

[16] Marcos Agustín Virreira. "Elementos y equipos eléctricos". En: *Elementos y equipos eléctricos* (1973).

[17] OpenAI. *Impact of Artificial Intelligence in Home Automation*. https://www.openai.com. 2024.

[18] OpenAI. *Machine Learning Applications in Home Automation*. https://www.openai.com. 2024.

[19] OpenAI. *Deep Learning in Home Automation*. https://www.openai.com. 2024.

[20] OpenAI. *Overview of Machine Learning*. https://www.openai.com. 2024.

[21] OpenAI. *Overview of Deep Learning*. https://www.openai.com. 2024.

[22] OpenAI. *Tiny Machine Learning: Bringing Machine Learning to Microcontrollers*. https://www.openai.com. 2024.

[23] OpenAI. *Convolutional Neural Networks for Movement Detection*. https://www.openai.com. 2024.

[24] OpenAI. *Recurrent Neural Networks for Movement Detection*. https://www.openai.com. 2024.

[25] OpenAI. *Long Short-Term Memory Networks for Movement Detection*. https://www.openai.com. 2024.

[26] OpenAI. *Temporal Convolutional Networks for Movement Detection*. https://www.openai.com. 2024.

[27] Marta López Enrique y García. "Aplicaciones de las redes neuronales y el deep learning a la modelización de series temporales financieras". En: *Redalyc* (2020). URL: https://www.redalyc.org/journal/5537/553768213002/html/.

[28] OpenAI. *Neural Network Input Layer*. https://www.openai.com. 2024.

[29] OpenAI. *Neural Network Hidden Layers*. https://www.openai.com. 2024.

[30] OpenAI. *Neural Network Output Layer*. https://www.openai.com. 2024.

[31] Autor Desconocido. *Modelo Simplificado de una Red Neuronal Artificial*. https://www.ejemplo.com/nn-image. 2024.

[32] OpenAI. *Weighted Connections in Neural Networks*. https://www.openai.com. 2024.

[33] OpenAI. *Cost Function in Neural Networks*. https://www.openai.com. 2024.

[34] OpenAI. *Optimization Algorithms in Neural Networks*. https://www.openai.com. 2024.

[35] IBM. *El modelo de redes neuronales*. https://www.ibm.com/docs/es/spss-modeler/saas?topic=networks-neural-model. Accedido: 2024-05-21. 2024.

[36] Author Name. "Recognition of Human Actions for Detection of Accidents in Intelligent Transportation Systems". En: *Journal of Electrical Systems and Information Technology* (2020). URL: https://www.sciencedirect.com/science/article/pii/S2605073020300298.

[37] Feng Yi. "Human Motion Recognition of Knitted Flexible Sensor in Walking Cycle". En: *Sensors* (2020). URL: https://www.researchgate.net/publication/338111710_Human_Motion_Recognition_of_Knitted_Flexible_Sensor_in_Walking_Cycle.

[38] Yanjinlkham Myagmar-Ochir y Wooseong Kim. "A Survey of Video Surveillance Systems in Smart City". En: *Electronics* 12.17 (2023), pág. 3567. URL: https://www.mdpi.com/2079-9292/12/17/3567.

[39] Nivian Home. *Cámaras de vigilancia WiFi para evitar robos en el hogar*. 2023. URL: https://www.nivianhome.com/es/camaras-vigilancia-wifi-evitar-robos-hogar/.

[40] Yanjinlkham Myagmar-Ochir y Wooseong Kim. "Human Action Recognition: A Taxonomy-Based Survey, Updates, and Opportunities". En: *Sensors* 23.4 (2023), pág. 2182. URL: https://www.mdpi.com/1424-8220/23/4/2182.

[41] Aiudo. *Caídas en ancianos: dispositivos para detectarlas*. 2023. URL: https://servicios.aiudo.es/teleasistencia/caidas-ancianos-dispositivos-para-detectarlas/.

[42] Jesús Robles-Serrano et al. "A review on action recognition for accident detection in smart city transportation systems". En: *Journal of Electrical Systems and Information Technology* (2023). URL: https://jesit.springeropen.com/articles/10.1186/s43067-023-00124-y.

[43] Interempresas. *Equipos de protección individual inteligentes: la protección del futuro*. 2023. URL: https://www.interempresas.net/Proteccion-laboral/Articulos/312165-Equipos-de-proteccion-individual-inteligentes-la-proteccion-del-futuro.html.

[44] Arduino. *Arduino - Home*. Accessed: 2024-05-31. n.d. URL: https://www.arduino.cc/.

[45] PlatformIO. *Your Gateway to Embedded Software Development Excellence*. Accessed: 2024-05-31. n.d. URL: https://platformio.org/.

[46] Mbed. *Free open source IoT OS and development tools from Arm*. Accessed: 2024-05-31. n.d. URL: https://os.mbed.com/.

[47] Espressif Systems. *ESP-IDF Programming Guide*. Accessed: 2024-05-31. n.d. URL: https://docs.espressif.com/projects/esp-idf/en/stable/esp32/index.html.

[48] STMicroelectronics. *STM32Cube - STM32 ecosystem*. Accessed: 2024-05-31. n.d. URL: https://www.st.com/en/development-tools/stm32cube.html.

[49] Google. *TensorFlow Lite for Microcontrollers*. Accessed: 2024-05-31. n.d. URL: https://www.tensorflow.org/lite/microcontrollers.

[50] Edge Impulse. *Edge Impulse - The edge AI platform*. Accessed: 2024-05-31. n.d. URL: https://www.edgeimpulse.com/.

[51] Python Software Foundation. *Welcome to Python.org*. Accessed: 2024-05-31. n.d. URL: https://www.python.org/.

[52] Google. *TensorFlow*. Accessed: 2024-05-31. n.d. URL: https://www.tensorflow.org/.

[53] Neuton AI. *Neuton TinyML Platform*. Accessed: 2024-05-31. n.d. URL: https://neuton.ai/.

[54] Douglas C Montgomery, Elizabeth A Peck y Geoffrey G Vining. *Introduction to Linear Regression Analysis*. John Wiley & Sons, 2021.

[55] Trevor Hastie, Robert Tibshirani y Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media, 2009.

[56] la enciclopedia libre Wikipedia. *Regresión lineal*. Accedido el 21 de junio de 2024. 2024. URL: https://es.wikipedia.org/wiki/Regresi%C3%B3n_lineal#/media/Archivo:Linear_regression.svg.

[57] *Nicla Sense ME — Arduino Official Store*. URL: https://store.arduino.cc/products/nicla-sense-me?_gl=1*23alki*_ga*Nzg2MjE2MjIwLjE3MTc2NDM5NDE.*_ga_NEXN8H46L5*MTcxNzY0MzkOMC4xLjAuMTcxNzY0MzkONS4wLjAuMTI2MzAzMTE5Mw..*_fplc*NGFkSGxrSWVWUzREU2NObFZRY29UZ1M1ZXdVRHclMkJORUhmZWVTancxYzlXbiUyRjV0ZW1QenM0MlQ1WFgyZnVlYklFMm40aDZzMVFMZzd5OE1seHF5UzlPJTJCZDZ4dUJZT1k5Z2O0MVElMkZMV2E3ajJ6SGNEaGt1UXh6cndscEI3VFElM0QlM0Q.*_gcl_au*MjA3OTA1MzQ3Mi4xNzE3NjQzOTQ2.

[58] *Model Settings*. 2019. URL: https://neuton.ai/st/99-model-settings.html.